

FAST ALGORITHMS FOR  
SMOOTH AND MONOTONE COVARIANCE MATRIX ESTIMATION

by

Adrian Aycañ Corum

Submitted to the Graduate School of Engineering and Natural Sciences  
in partial fulfillment of the requirements for the degree of  
Master of Science in Electronics Engineering

Sabanci University  
August 2012


FAST ALGORITHMS FOR  
SMOOTH AND MONOTONE COVARIANCE MATRIX ESTIMATION

APPROVED BY:

Assist. Prof. Dr. Müjdat Çetin  
(Thesis Supervisor)



Prof. Dr. Ş. İlker Birbil



Assoc. Prof. Dr. Kerem Bülbül



Assist. Prof. Dr. Semih Onur Sezer



Assoc. Prof. Dr. Koray Deniz Şimşek



DATE OF APPROVAL: July 31, 2012

©Adrian Aycan Corum 2012

All Rights Reserved

*to my beloved wife Beril,  
my caring grandmother Cavidan,  
and living memory of my role model and grandfather Yılmaz*

## ACKNOWLEDGMENTS

The presented work would not have been achieved without the help and support of a range of people and I would like to take the opportunity to express them my gratitude.

I would like to thank to my thesis supervisor Prof. Müjdat Çetin for his endless in-depth professional guidance beginning from even before my master's, his immense technical contribution to this thesis, and his effort to make my life during my master's as easy as possible. I would also like to express my gratitude to Müjdat for his great care regarding my personal life on many occasions, which has been one of the key and exemplary factors for establishing an effective advisor-student relationship. I would like to thank Dr. Dmitry Malioutov, who has helped me much more than a thesis co-supervisor would but whom unfortunately I cannot include formally as my thesis co-supervisor due to the regulations I need to follow, for his never-ending hands-on collaboration on every practical and theoretical aspect of the thesis. I am extremely grateful to Müjdat and Dmitry for their patience and support that expand well beyond their responsibilities as research advisors.

I would like to thank Prof. İlker Birbil, Prof. Kerem Bülbül, Prof. Semih Sezer, and Prof. Koray Şimşek for serving on my thesis committee and for their invaluable suggestions for improvement of the thesis. I would also like to thank Prof. Hakan Erdoğan for attending my thesis defense.

I would like to acknowledge The Scientific and Technological Research Council of Turkey (TÜBİTAK) for providing financial support for my master's.

I am deeply indebted to my wife Beril. Her support, encouragement, and tolerance was in the end what made this thesis possible. I would like to thank her not only for her unconditional love, continuous care, trust, and passion, but also for her sacrifice, understanding, and patience ever since we have been together. She is the one who has made me see how to develop a taste for life by preventing me from falling into one of the abysses of sophisticated mediocrity and distraction of our modern world. We will make our dreams come true together.

I would like to thank my grandmother Cavidan and my grandfather Yılmaz, who relentlessly gave their all effort and spent a very significant portion of their lives to raise me the best way they could and to make a person as self-confident as possible out of me by not only providing limitless resources but also conveying the practical and philosophical importance of these resources, including those which are the most scarce in this world yet the most vital to a soul, such as boundless amount of love as well as the best education they could sustain. They have proved this attitude

countless times, including sincere willingness and encouragement of my grandmother for me to do my PhD so far away from her, despite especially her age. I will always work towards developing their ideals, especially those of my grandfather's, who was a great philosopher and who will always continue to be my role model.

I, of course, would like to use this chance also to thank not only my mother Hülya and brother Kaan, but also my mother-in-law Meryem, father-in-law Gürcan, and brother-in-law Ege for their continuous understanding and support. Another thanks to my wife for making me a part of a family of such invaluable people.

# FAST ALGORITHMS FOR SMOOTH AND MONOTONE COVARIANCE MATRIX ESTIMATION

Adrian Aycan Corum

Electronics Engineering, MS Thesis, 2012

Thesis Supervisor: Assistant Prof. Dr. Müjdat Çetin

Keywords: financial risk management, optimal first-order methods, covariance matrix estimation, semidefinite programming

## Abstract

In this thesis the problem of interest is, within the setting of financial risk management, covariance matrix estimation from limited number of high dimensional independent identically distributed (i.i.d.) multivariate samples when the random variables of interest have a natural spatial indexing along a low-dimensional manifold, e.g., along a line.

Sample covariance matrix estimate is fraught with peril in this context. A variety of approaches to improve the covariance estimates have been developed by exploiting knowledge of structure in the data, which, however, in general impose very strict structure.

We instead exploit another formulation which assumes that the covariance matrix is smooth and monotone with respect to the spatial indexing. Originally the formulation is derived from the estimation problem within a convex-optimization framework, and the resulting semidefinite-programming problem (SDP) is solved by an interior-point method (IPM). However, solving SDP via an IPM can become unduly computationally expensive for large covariance matrices.

Motivated by this observation, this thesis develops highly efficient first-order solvers for smooth and monotone covariance matrix estimation. We propose two types of solvers for covariance matrix estimation: first based on projected gradients, and then based on recently developed optimal first order methods. Given such numerical algorithms, we present a comprehensive experimental analysis. We first demonstrate the benefits of imposing smoothness and monotonicity constraints in covariance matrix estimation in a number of scenarios, involving limited, missing,

and asynchronous data. We then demonstrate the potential computational benefits offered by first order methods through a detailed comparison to solution of the problem via IPMs.



# TEKDÜZE VE PÜRÜZSÜZ ORTAK DEĞİŞİNTİ MATRİSİ KESTİRİMİ İÇİN HIZLI ALGORİTMALAR

Adrian Aycan Corum

Elektronik Mühendisliği, Yüksek Lisans Tezi, 2012

Tez Danışmanı: Yard. Doç. Dr. Müjdat Çetin

Anahtar Sözcükler: finansal risk yönetimi, optimal birinci derece yöntemler, ortak değişinti matrisi kestirimi, yarı kesin programlama

## Özet

Bu tezin üzerine eğildiği problem, finansal risk yönetimi bağlamında, bağımsız özdeş dağılımlara sahip sınırlı sayıda ve yüksek boyutlu çok-değişkenli örneklerden söz konusu rasgele değişkenlerin düşük-boyutlu bir çok-katmanlı (örneğin bir doğru) boyunca kendiliğinden uzamsal bir dizilimi olması koşulu altında ortak değişinti matrisi kestirimidir.

Örnekleme ortak değişinti matrisi kestirimi yaklaşımı söz konusu çerçevede içinde birçok risk barındırmaktadır. Ortak değişinti matrisi kestirimlerini geliştirmek amacıyla verinin yapısı hakkındaki bilgilerden faydalanan birtakım yaklaşımlar geliştirilmiş olsa da genelde hepsi çok katı yapılar empoze etmektedirler.

Bu tezde ise, ortak değişinti matrisinin bahsi geçen uzamsal dizinlemeye göre tekdüze ve pürüzsüz olduğunu varsayan farklı bir formülasyondan yararlanılmaktadır. Bu formülasyon orijinal olarak söz konusu kestirim probleminden dışbükey eniyileme çerçevesi dahilinde türetilmiş olup sonucunda elde edilen yarı kesin programlama problemi (SDP) bir dahili nokta yöntemi (IPM) ile çözülmektedir. Fakat bir IPM'ni SDP ile çözmek büyük ortak değişinti matrisleri için hesaplama bakımından aşırı masraflı olabilir.

Bu gözlemden harekete geçerek, bu tezde tekdüze ve pürüzsüz ortak değişinti matrisi kestirimi için yüksek verimli birinci derece çözücüler geliştirmekteyiz. İlki izdüşümsel gradyanlar, ikincisi de yeni geliştirilmiş optimal birinci derece yöntemler üzerine dayalı olmak üzere ortak değişinti matrisi kestirimi için iki çeşit çözücü önermekteyiz. Bu sayısal algoritmalar ile kapsamlı bir deneysel analiz sunmaktayız. Öncelikle verilerin sınırlı, eksik, veya zamanuyumsuz olduğu durumlarda ortak değişinti matrisi kestirimi üzerinde tekdüzelik ve pürüzsüzlük kısıtlarını uygu-

lamannın faydalarını göstermekteyiz. Sonrasında birinci derece yöntemlerimizin olası hesapsal faydalarını problemin IPM ile çözümümüyle ayrıntılı bir şekilde karşılaştırarak göstermekteyiz.

# Table of Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Motivation . . . . .	1
1.2	Problem Definition and State of the Art . . . . .	2
1.3	Contributions of the Thesis . . . . .	4
1.4	Thesis Organization . . . . .	5
<b>2</b>	<b>Background</b>	<b>7</b>
2.1	Sample Covariance Matrix Estimate . . . . .	7
2.2	Principal Component Analysis (PCA) and Factor Analysis (FA) . . .	9
2.3	Sparsity of the Information Matrix . . . . .	10
2.4	Parametric Models . . . . .	12
2.5	Other methods . . . . .	13
<b>3</b>	<b>Smooth and Monotone Formulation for Covariance Matrix Estimation</b>	<b>15</b>
3.1	Motivation . . . . .	15
3.2	Formulation and Solution through IPM . . . . .	16
3.2.1	Formulation . . . . .	16
3.2.2	Solution through IPM . . . . .	19
3.3	Experimental Results . . . . .	21
3.3.1	Term-structure Modeling . . . . .	21
3.3.2	Experiments with a Known Underlying Covariance Matrix . .	23
3.3.3	Missing data . . . . .	24
3.3.4	Out-of-sample Covariance Prediction . . . . .	25
3.3.5	Spectral Correction . . . . .	27
<b>4</b>	<b>Fast Algorithms for Smooth and Monotone Covariance Matrix Estimation</b>	<b>31</b>
4.1	Original Gradient Projection Method Revisited . . . . .	32
4.2	Dual Projected Gradient Solution for the Monotone Problem . . . . .	33

4.3	Dual Projected Coordinate Descent Solution for the Smooth and Monotone Problem . . . . .	37
4.3.1	Exercise: Dual Projected Coordinate Descent Solution for the LSCAP . . . . .	38
4.3.2	Solution for the Smooth and Monotone Problem . . . . .	41
4.4	Optimal First Order Methods with FISTA . . . . .	52
4.4.1	FISTA Revisited . . . . .	52
4.4.2	Optimal First Order Method for the Monotone Problem . . . . .	53
4.4.3	Optimal First Order Method for the Smooth and Monotone Problem . . . . .	55
4.5	Experiments and Results . . . . .	57
4.5.1	The Monotone Problem . . . . .	58
4.5.2	The Smooth and Monotone Problem . . . . .	64
<b>5</b>	<b>Conclusion</b>	<b>67</b>
5.1	Summary of the Thesis and of the Contributions . . . . .	67
5.2	Future Work . . . . .	68
5.2.1	Application . . . . .	68
5.2.2	Analysis . . . . .	69
5.2.3	Formulation . . . . .	69
<b>A</b>	<b>Mathematical Preliminaries</b>	<b>71</b>
A.1	Convex Sets and Cones, and Relation to Positive Semidefiniteness and Generalized Inequalities . . . . .	71
A.2	Convex Optimization . . . . .	74
A.3	Duality . . . . .	76
A.4	Matrix Calculus . . . . .	80
<b>B</b>	<b>Derivations for Subsection 4.3.2</b>	<b>83</b>
	<b>Bibliography</b>	<b>87</b>

# List of Figures

2.1	Marcenko-Pastur law and the sample eigenvalue spectrum from $\mathcal{N}(0, \mathcal{I})$ . True eigenvalues are all 1. . . . .	8
2.2	(a) True spectrum, $N = 150$ . (b) Spectrum from sample covariance, $T = 500$ (c) $T = 10000$ . . . . .	8
3.1	Term-rate covariances: (a) true (b) sample estimate. . . . .	17
3.2	Covariance regularization: (a) monotone (b) monotone and smooth. . .	18
3.3	Sample ED curves, linearly interpolated. . . . .	22
3.4	Alternative estimates: (a) MRF (b) PCA. . . . .	23
3.5	Errors of sample covariance, monotonic, smooth, GM and PCA estimates. . . . .	24
3.6	(a) Sample covariance with missing data. (b) Recovered smooth-monotone covariance. . . . .	25
3.7	(a) Frobenius error over running windows. (b) Average Frobenius errors as a function of training window length. (c) Average percentage improvement of forecast error from $P_{SM}$ over $\hat{P}$ . . . . .	26
3.8	What percent $P_{SM}$ outperforms $\hat{P}$ (on average) (a) for $T_{TR}$ vs $k \triangleq T_{TEST}/T_{TR}$ (b) for $T_{TR}$ vs $T_{TEST}$ . . . . .	27
3.9	Projection onto the p.s.d. cone vs. smooth and monotone solution. . .	28
3.10	(a) True, sample, and smooth-monotone eigen-spectra. (b) detail (c) log-scale of true and smooth-monotone spectra . . . . .	29
4.1	Convergence characteristics ( $\ P_{\text{grad},k} - P_{\text{IPM}}^*\ _F$ at each iteration $k$ ) of Algorithm 1 when $N = 15$ for step sizes (a) $\alpha = 2/L = 2/896$ (b) $\alpha = 1/8$ . . . . .	58
4.2	Characteristics of Algorithm 1 when $N = 15$ for convergence to $P_{\text{IPM}}^*$ and $P_{\text{grad}}^*$ ( $\ P_{\text{grad},k} - P_{\text{IPM}}^*\ _F$ and $\ P_{\text{grad},k} - P_{\text{grad}}^*\ _F$ at each iteration $k$ ). . . . .	59
4.3	Convergence characteristics ( $\ P_{\text{grad}+\text{FISTA},k} - P_{\text{IPM}}^*\ _F$ at each iteration $k$ ) of Algorithm 6 when $N = 15$ for step sizes (a) $\alpha = 1/L = 1/896$ (b) $\alpha = 1/12$ . . . . .	60

4.4	When $N = 15$ and for step sizes (for curves from left to right) $\alpha = 1/12, 1/100, 1/896, 1/1000$ , and $1/10000$ , characteristics of Algorithm 6 regarding (a) convergence (b) norm change between the covariance matrix iterates. . . . .	61
4.5	Characteristics of Algorithm 6 when $N = 15$ for convergence to $P_{\text{IPM}}^*$ and $P_{\text{grad}}^*$ ( $\ P_{\text{grad+FISTA},k} - P_{\text{IPM}}^*\ _F$ and $\ P_{\text{grad+FISTA},k} - P_{\text{grad}}^*\ _F$ at each iteration $k$ ). . . . .	62
4.6	Different performances of Algorithm 6 (blue curves) with respect to Algorithm 1 (red curves) for different samples: Algorithm 6 converges to $P_{\text{IPM}}^*$ (a) faster than (b) as fast as (c) slower than Algorithm 1. . .	62
4.7	Median and 25 <sup>th</sup> -75 <sup>th</sup> percentile time it takes for IPM (black curves), Algorithm 1 (red curves), and Algorithm 6 (blue curves) to converge to $P_{\text{IPM}}^*$ for $N$ from 10 to 50. . . . .	63
4.8	Median time it takes for Algorithm 1 (red curves) and Algorithm 6 (blue curves) to reach to $10^{-x}$ proximity of $P_{\text{IPM}}^*$ for (a) $x = 1$ (b) $x = 3$ (c) $x = 5$ , including 25 <sup>th</sup> -75 <sup>th</sup> percentiles for (a) and (b), for $N$ from 10 to 50. . . . .	64
4.9	Smooth-monotone: median and 25 <sup>th</sup> -75 <sup>th</sup> percentile time it takes for IPM (black curves), Algorithm 4 (red curves), and Algorithm 7 (blue curves) to converge to $P_{\text{IPM}}^*$ for $N$ from 10 to 50. . . . .	65
4.10	Smooth-monotone: median time it takes for Algorithm 4 (red curves) and Algorithm 7 (blue curves) to reach to $10^{-x}$ proximity of $P_{\text{IPM}}^*$ for (a) $x = 1$ (b) $x = 3$ (c) $x = 6$ , including 25 <sup>th</sup> -75 <sup>th</sup> percentiles for (a) and (b), for $N$ from 10 to 50. . . . .	66

# List of Tables

2.1	Summary of several commonly-used covariance functions. . . . .	12
-----	--	----





# Chapter 1

## Introduction

In this thesis the problem of interest is covariance matrix estimation from limited number of high dimensional independent identically distributed (i.i.d.) multivariate samples when individual random variables of the random vector have a natural spatial indexing along a low-dimensional manifold, e.g., along a line. For this problem we take as basis the smooth-monotone estimation formulation that allows all the elements of the covariance matrix to be treated as separate parameters, but requires the covariance function to be smooth and monotone with respect to this indexing. The primary aim of the thesis is to develop highly efficient first-order solvers for this smooth-monotone formulation. The secondary aim is to present extensive simulations of (1) the developed first order solvers, which are based on this formulation, regarding their computational benefits and of (2) the smooth and monotone covariance estimation formulation regarding its accuracy.

### 1.1 Motivation

Modeling joint statistical dependence among a collection of random variables is one of the central problems in statistics, machine learning and engineering. A recent trend in these areas has been the analysis of high-dimensional models where the number of parameters may be comparable or higher than the number of available data points. This is because lately many of the applied problems have grown increasingly high-dimensional, making these models not only of considerable theoretical interest but also of practical importance in applications such as financial portfolio management in financial engineering, pathway discovery in gene-regulatory networks, computer vision, and many others in numerous other areas, including social networks and brain and cognitive science. The covariance matrix remains one of the the most popular tools for capturing the strength of association among the variables.

However, even estimating the covariance matrix from i.i.d. multivariate samples

in high-dimensions is very challenging when one is faced with limited data. It is well-known that the sample covariance matrix is fraught with peril in this setting. In particular, the inconsistency of its eigenvalue spectrum has grave implications for financial risk management when the sample covariance estimate is used within the Markowitz portfolio optimization framework [32]. A variety of approaches to improve the covariance estimates have been developed by exploiting knowledge of structure in the data, including low-rank models (principal component [2,20] and factor analysis [21]), sparse inverse covariance [6], and parametric models [13]. Although these models have successful practical applications in their respective domains, in general they manage to reduce the required number of samples by imposing very strict structure.

The limitations of the mentioned models leave an open end to study other formulations assuming a different prior that does not directly limit the number of parameters but still reduces the complexity of the space of their joint configurations. These more flexible formulations may be constructed in an optimization framework, Once this framework is exploited, then the speed and efficiency of the algorithm used for solving the formulation become an important practical aspect.

## 1.2 Problem Definition and State of the Art

The problem we want to solve, as mentioned at the beginning of this chapter, is covariance matrix estimation from limited number of high dimensional *i.i.d.* multivariate samples when individual random variables of the random vector have a natural spatial indexing along a low-dimensional manifold, e.g., along a line (To visualize, one may think of, for example, acoustic measurements at microphones along a linear sensor array. Note that the indexing is spatial). We will consider this problem within the setting of financial risk management, where the Gaussian model of risk is the underlying assumption of the Markowitz portfolio optimization framework [34] widely used in the industry. To be specific, we will consider the applications of this problem in interest rate term-structure modeling (Again for example, this time one may think of daily changes in prices of Eurodollar futures contract with expiration  $k$  quarters (multiples of 3 months) from the present. In this case the indexing is with respect to  $k$ , again a spatial indexing.). When large collection of financial instruments are modeled in this setting, these instruments cause this setting to be not only high dimensional due to the large size of the collection but also interpretable as scarce data due to the fact that typically the data are quickly evolving, rendering the old samples unreliable and hence limiting one to use only recent samples.

Both statisticians [32] and practitioners have realized that using the sample co-

variance matrix estimate is a disastrous choice when one is modeling large collection of financial instruments in the setting of financial risk management. The sample covariance matrix with scarce data produces an inconsistent estimate of the eigenvalue spectrum, and when it is used to create optimized portfolios the solution tends to prefer those instruments which have underestimated risk. The end-result could be a vast understatement of risk of the Markowitz portfolio. Therefore, although the sample covariance matrix estimate is unbiased and consistent in the high-sample regime, it requires strong regularization in the high-dimensional scarce data setting. Progress can be made by relying on prior knowledge of the structure of the data. Here, it is crucial to describe such a structured model carefully so that the complexity of the parameter space can be simplified dramatically without adding significant bias. However, in general, the assumptions for existing methods tend to impose too strong of a structure, such as in the models briefly mentioned in the following.

A widely used assumption stipulates that the data lie on or near a low-dimensional manifold, in particular a linear manifold. For covariance matrix estimation this translates into principal component analysis (PCA) or factor analysis (FA) [35]. The covariance matrix is assumed to be low-rank plus perhaps a diagonal noise term, thus reducing the number of parameters from  $N^2$  to  $NK$ , where  $N$  is the dimension and  $K$  is the assumed rank. Another approach relies on the sparsity of the *information matrix*, i.e., the inverse of the covariance matrix. This is known as a *covariance selection model* in statistics and as *Gaussian graphical model* or *Gaussian Markov Random Field (MRF)* in machine learning [7, 19]. The pattern of nonzero elements of the information matrix captures the conditional independence structure, with the number of such elements often assumed to be bounded by a small constant  $K$ , again reducing the total number of parameters to  $NK$ . *Banded covariance matrices* that allow only a certain number of nonzero diagonals (bands) have been investigated in [30]. Parametric models provide another popular regularization choice by assuming that entries of the covariance matrix follow some functional form: for example the  $i, j$ -th element  $P_{(i,j)}$  of the covariance may be assumed to decay exponentially or as a power-law with some notion of distance from  $i$  to  $j$ , e.g.,  $P_{(i,j)} \propto \exp(-d(i, j))$ . Gaussian Processes (GP) constitute a general framework for such models [13]. Shrinkage estimates [31] take a weighted combination of the sample covariance matrix and a strongly-regularized model (such as low-rank). While they do improve the expected mean-squared error, they do not add any new kind of structure. We note that all of the above models have very successful domains of applications, but in general they manage to reduce the required number of samples by imposing very strict structure.

In this thesis we instead use the formulation originally presented in the short paper [28] which investigates a different prior for random vectors indexed along a

low-dimensional manifold. This formulation allows all the elements of the covariance matrix to be treated as separate parameters, but requires the covariance function to be smooth and monotone (*isotonic*) with respect to this indexing, a natural assumption for a variety of problems including those in our setting of financial risk management, e.g., interest-rate risk modeling in financial engineering. While not directly limiting the number of parameters, the complexity of the space of their joint configurations is thus reduced: this is a regularization approach to covariance estimation.

Related approaches have been studied in nonparametric statistics for applications including monotone density and function estimation, spline smoothing, etc. [38]. Moreover, the smoothness of the covariance function has been mentioned in prior work: its importance was noted in [36], where smoothness of covariance functions via local-cosine bases expansions was used, and it was used as an assumption in [37] to efficiently approximate variances in large-scale Gaussian MRF models. However, in this thesis we are specifically interested not just in the diagonal of the covariance matrix or its near-diagonal elements, but rather in the whole covariance matrix, just like in [28], which also does not assume any MRF structure.

### 1.3 Contributions of the Thesis

We take the covariance matrix estimation approach in [28] as the basis of this thesis. Our first contribution, described in Section 3.3, is to demonstrate the application of this approach on a number of examples not only with limited data, but also with missing and asynchronous data after describing its extensions to problems with such data. With these extensions and experiments we show that it has the potential to provide more accurate covariance matrix estimates than existing methods and exhibits a desirable eigenvalue-spectrum correction effect.

A novel aspect of applying the approach in [28] is the inherent requirement of semipositive-definiteness, and in that paper the estimation problem was formulated as semidefinite programming (SDP) and solved via an interior-point method (IPM). However, solving SDP via an IPM can become unduly computationally expensive for large covariance matrices, as it involves computing the Hessian. This is the motivation behind the main contribution of this thesis, which appears in Chapter 4. We present an alternate perspective and develop optimal first-order methods for solving this optimization problem, especially with much larger covariance matrices. In our derivation we first adapt the projected gradient method of [26] and accelerate it following the ideas in [25].

Our final contribution, appearing in Section 4.5, is to demonstrate the compu-

tational benefits offered by the first order methods we develop and to provide a detailed comparison to solution of the problem via IPMs.

## 1.4 Thesis Organization

The remainder of this thesis is organized as follows.

■ **Chapter 2 – Background.** In this chapter, we first overview a number of existing covariance matrix estimation approaches for the low sample regime. Before starting this overview, we first explain and demonstrate the perils of using sample covariance matrix estimate in this setting, the simplest approach in covariance matrix estimation. In order to improve on the sample covariance matrix estimate, some kind of prior should be assumed. Therefore, the methods we present in the overview rely on prior knowledge of the structure of the data, which include principal component analysis and factor analysis, sparsity of the information matrix, and parametric models. We also mention some other relevant methods, i.e., banded approximation model and shrinkage estimate, at the end of the section. We then provide some mathematical preliminaries which will be of use in the thesis.

■ **Chapter 3 – Smooth and Monotone Formulation for Covariance Matrix Estimation.** This chapter contains the formulation of covariance estimation in [28] as an optimization problem involving a data fidelity term as well as constraints imposing smoothness and monotonicity of the covariance matrix. We first motivate this formulation in Section 3.1. Following, in Section 3.2, we formulate the estimation problem in a convex-optimization framework, and propose solving the resulting semidefinite-programming problem by an interior-point method. In Section 3.3, we make our first contribution by demonstrating the application of our approach on a number of examples with limited, missing and asynchronous data, and showing that it has the potential to provide more accurate covariance matrix estimates than existing methods, and exhibits a desirable eigenvalue-spectrum correction effect.

■ **Chapter 4 – Fast Algorithms for Smooth and Monotone Covariance Matrix Estimation.** Solving an SDP using an IPM as proposed in Chapter 3 can become unduly computationally expensive for large covariance matrices, as it involves computing the Hessian. In this chapter we make our main contribution through Sections 4.2 - 4.4 by developing optimal first-order methods for solving this optimization problem. In our derivation we first adapt the projected gradient method of [26] and accelerate it following the ideas in [25]. Therefore, first of all, in Section 4.1 we start with revisiting the original gradient projection method developed by Boyd and Xiao [26]. After that section we start developing our ideas to produce faster algorithms. For pedagogical reasons, we first develop these ideas for the special

case of our problem which contains monotonicity constraints only. Therefore, we start with describing a dual first-order method based on gradient projection [26] for our monotone problem in Section 4.2. Following, in Section 4.3, we develop a dual projected coordinate descent solution for our smooth and monotone problem, which is also a first-order method, inspired from the method developed for the monotone problem. In Section 4.4 we develop even faster versions first for our monotone problem and then for our smooth and monotone problem using FISTA, i.e., the optimal first order ideas of [25]. Finally, we present our final contribution in Section 4.5 as a detailed experimental analysis demonstrating the computational benefits offered by the algorithm we develop in this chapter.

■ **Chapter 5 – Conclusion.** This chapter provides concluding remarks and summarizes the main contributions of this thesis. Several extensions to the ideas presented here are discussed, with a number of suggestions for further research.

# Chapter 2

## Background

In this chapter we overview a number of existing covariance matrix estimation approaches for low sample regime. Before starting this overview, we first explain and demonstrate in Section 2.1 the perils of using in this setting sample covariance matrix estimate, the simplest approach in covariance matrix estimation. In order to improve on the sample covariance matrix estimate, some kind of prior should be assumed. Therefore, the methods we present in the following overview rely on prior knowledge of the structure of the data. We explain principal component analysis and factor analysis in Section 2.2, sparsity of the information matrix in Section 2.3, and parametric models in Section 2.4, mentioning some other relevant methods as well in Section 2.5. The important point here from our perspective is that all of these methods have successful practical applications in their domains but also limitations since in general they manage to reduce the required number of samples by imposing very strict structure.

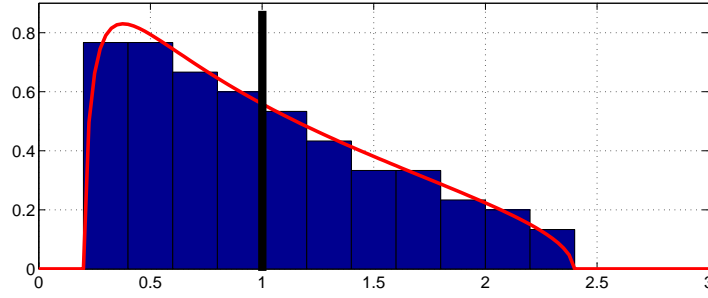
### 2.1 Sample Covariance Matrix Estimate

The sample covariance matrix

$$\hat{P} \triangleq \frac{1}{T} \sum_{i=1}^N \mathbf{x}(t_i) \mathbf{x}(t_i)^T \quad (2.1)$$

is an unbiased and consistent estimate in the high-sample regime,  $T/N \rightarrow \infty$ , but with scarce data it has well-documented failures [32]. In particular, the eigenvalue spectrum is biased with  $T/N$  held fixed, as  $T \rightarrow \infty$ . This creates a significant problem when sample covariance is used for risk-modeling in Markowitz portfolios: the optimized portfolio tends to be aligned with the most underestimated components of risk, and with less weight in over-estimated ones, causing severe overall risk underestimates [32].

Consider the eigenvalues of the sample covariance matrix obtained from  $T$  i.i.d. samples from the multivariate standard normal  $\mathcal{N}(0, I)$  in  $N$ -dimensions, with  $\rho =$

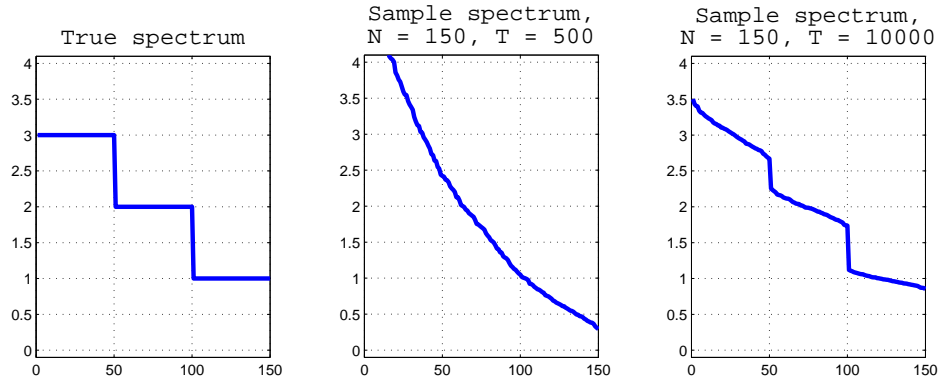


**Figure 2.1.** Marcenko-Pastur law and the sample eigenvalue spectrum from  $\mathcal{N}(0, \mathcal{I})$ . True eigenvalues are all 1.

$N/T$ . The true eigenvalues are all 1. The sample eigenvalue spectrum asymptotically follows the Marcenko-Pastur law<sup>1</sup> illustrated in Figure 2.1:

$$f_p(x) = \frac{1}{2\pi} \frac{\sqrt{(y_+ - x)(x - y_-)}}{x},$$

where  $y_{\pm} = (1 \pm \sqrt{\rho})^2$ . Hence, the smallest eigenvalue (corresponding to the direction which allegedly has the least risk) is a severe underestimate of its true value 1 for small and moderate  $T$ .



**Figure 2.2.** (a) True spectrum,  $N = 150$ . (b) Spectrum from sample covariance,  $T = 500$  (c)  $T = 10000$ .

To illustrate the gravity of the problem we consider a numerical example with  $N = 150$  in Figure 2.2. The true covariance is taken to have a blocky eigen-spectrum in plot (a). With  $T = 500$  samples the sample covariance matrix produces a very smoothed-out eigen-spectrum in plot (b). Even with  $T = 10,000$  in plot (c), we still get a very distorted spectrum! (We will see in Subsection 3.3.5 that our approach provides a much superior spectrum estimate than the sample covariance. This can greatly help to mitigate the problem of bad risk forecasts in optimized portfolios based on interest-rate curves.)

<sup>1</sup>The law is asymptotic, but empirically it also describes the finite sample cases well.



## 2.2 Principal Component Analysis (PCA) and Factor Analysis (FA)

A widely used assumption for covariance matrix estimation is that the data lie close to a low-dimensional subspace, which, for covariance estimation may translate into PCA or FA [20]. In this respect, the motivation for PCA is to find a lower dimensional subspace that captures most of the variance and this is done with eigen-decomposition of the sample covariance matrix  $\hat{P}$ . Each principal component has a corresponding eigenvector and eigenvalue, constituting

$$\hat{P} = Q\Lambda Q^T = \sum_{i=1}^N \lambda_i v_i v_i^T,$$

where  $Q$  is  $N \times N$  matrix of eigenvectors  $v_i$  and where  $\Lambda$  is a diagonal  $N \times N$  matrix of eigenvalues  $\lambda_i$ , each corresponding to  $v_i$ . These principal components are ordered with respect to the information they carry about the matrix (their eigenvalues in decreasing order to be specific). The assumption used in PCA, that most of the variance in the real covariance matrix  $P$  lies in a  $K$ -dimensional subspace where  $K < N$  [2], leads to the inference that only first  $K$  principal components carry worthwhile information about  $P$  that hence an insignificant amount of error would be made by attesting low-rank property to  $P$ , i.e., a rank of  $K < N$ . This reduces the number of parameters from  $N^2$  to  $NK$ , and the covariance matrix estimate for PCA becomes

$$P_{\text{PCA}}^* = \sum_{i=1}^K \lambda_i v_i v_i^T, \quad (2.2)$$

FA, or EFA (Exploratory Factor Analysis), is a related technique that assumes that  $P$  can be decomposed as a sum of a rank- $K$  matrix and a diagonal matrix and which reduces the number of parameters to approximately again  $NK$ . FA is a latent variable model [21] according to which the random vector  $\mathbf{x} = (x_1, x_2, \dots, x_N)^T$  of size  $N$  is actually determined by a smaller random vector  $\mathbf{s} = (s_1, \dots, s_K)^T$ , which is of size  $K < N$  and where  $s_j$ 's are independent, with a linear relation  $A$  plus an independent zero-mean random error vector  $\mathbf{e} = (\varepsilon_1, \varepsilon_2, \dots, \varepsilon_N)^T$ :

$$\mathbf{x} = A\mathbf{s} + \mathbf{e},$$

i.e., each  $x_i$  is determined by a linear combination of the  $s_j$ 's plus an independent zero-mean error term  $\varepsilon_i$ :

$$x_i = [A]_{(i,1)}s_1 + \dots + [A]_{(i,K)}s_K + \varepsilon_i, \quad \text{for all } i = 1, \dots, N.$$

This assumption reduces  $P$  to a sum of a diagonal term  $D_\varepsilon$  (dictated by the structure of independent errors) and a matrix  $AA^T$  of rank  $K$  which results from

the linear transformation of  $\mathbf{s}$ , reducing as well the number of parameters to  $N + NK$ :

$$P_{\text{FA}} = P_X = AP_S A^T + P_\epsilon = AA^T + D_\epsilon, \quad (2.3)$$

where the result  $PS = I$  by the assumption of independence of  $s_j$ 's is exploited.

In all implementations of FA, eigen-decomposition of  $\hat{P}$  and its first  $K$  factors are involved. The rest of the implementation, however, depends on how the diagonal term is structured. In the simplest case the diagonal term is just the identity matrix multiplied by a common standard deviation  $\sigma$ , then the problem reduces to just finding  $\sigma$  and the factors. In that case, if in addition  $N - K$  eigenvalues of  $\hat{P}$  are equal to  $\sigma^2$ , both of these values can be found by the mentioned eigen-decomposition, without the need for iteration [4,5]. Otherwise and also in the case that the diagonal term is allowed to have arbitrary positive standard deviations, the first  $K$  factors and the diagonal term are fitted iteratively.

It is important to note here that although PCA and FA are related models, PCA is strictly a dimensionality reduction technique, while FA often comes from assuming a generative model. One of most important resulting distinction is that FA treats the covariance and variance separately (with  $AA^T$  and  $D_\epsilon$ , respectively) whereas PCA treats them identically [3, 21].

There are of course cases where these low-rank models work great, but their corresponding assumptions may not be always valid.

## 2.3 Sparsity of the Information Matrix

An information matrix is by definition the inverse of a covariance matrix, and there is a class of models using the sparsity of the information matrix. This method is again used to reduce the total number of parameters to  $NK$ , by usually bounding the number of nonzero elements of the information matrix by a constant  $K$ . The reason for limiting the number of these elements is that the structure of conditional independence is reflected by the pattern of these elements. The idea of sparsity of the information matrix is realized under different names in separate disciplines. It is known as a *covariance selection model* in statistics and as *Gaussian graphical model* or *Gaussian Markov Random Field (GMRF)* in machine learning.

The development in the rest of this subsection follows that in [6]. A graphical model in this context represents the joint probability distribution, and the aim in using this model is to decompose the distribution into products of simple local functions that only depend on small subsets of variables. In the graph, a random variable is associated with each vertex, and the edges or cliques represent the local functions. In areas such as computer vision and genomics the graph may have a direct physical

meaning such as the embodiment of the similarity among nearby pixels or biological interactions among genes as the edges and vertices respectively, in others it may not have such a meaning and may be heavily used for the goal of efficiency. The important point here is that what captures the conditional independence structure among the random variables is this graph, and it makes the concept of graphical models very effective.

To be able to use a graphical model in an application, choosing one of its special cases is essential. One of the options is to restrict the model simultaneously to undirected graphs, i.e. *Markov random fields (MRF)*, and to variables with jointly Gaussian distribution. With these two restrictions we are now looking at what is called GMRF (or also Gaussian graphical model - GGM), while this model has been earlier exploited with the name covariance selection model under the discipline of statistics [7, 8]. These models are again used in numerous areas, including machine learning and optimization, especially for quadratic problems [10, 11, 12].

What makes the GGM so special among the graphical models is that the mentioned structure of conditional independence among certain sets of variables can be simply and explicitly obtained from the inverse covariance matrix  $J \triangleq P^{-1}$ , i.e. the information matrix. The explicitness is the key: When  $[J]_{(i,j)} = 0$ , this corresponds to the edge  $(i, j)$  being missing from the graph, and this leads to the direct inference that  $x_i$  and  $x_j$  are conditionally independent given the rest of the variables. This relates the sparsity of  $J$  to Markov structure, and major advantage of using  $J$  leads to parameterizing the Gaussian probability density heavily in terms of  $J$ , an easy modification since the original density already includes  $P^{-1}$  in the formulation:

$$\begin{aligned} p(x) &\propto \exp\left(-\frac{1}{2}(x-\mu_x)^T P^{-1}(x-\mu_x)\right) = \exp\left(-\frac{1}{2}(x-\mu_x)^T J(x-\mu_x)\right) \\ &\propto \exp\left(-\frac{1}{2}x^T Jx + (J\mu)^T x\right) \end{aligned}$$

The way GGM reduces the total number of parameters in the model is to make the prior model  $p(x)$  specify a sparse  $J$  matrix. Typically marginal densities (described by marginal means and variances) at each node are used to compute the MAP (max a-posteriori) estimate of  $x$  after adding the measurements on top of the prior density and hence forming the posterior density.

One major downside of GGM is that its complexity is dominated by the matrix inversion operation which has a cubic complexity in the number of the variables, except for when there is an assumption that graph is very sparse or near-tree structured which is correspondingly a very heavy restriction. Another downside is of course the direct restriction of parameters as in PCA and FA. Nevertheless, however, the method has always gathered a great attention and proposals for learning methods to fit sparse inverse covariance matrices to the data (i.e. learning a GMRF) have been one of the centers of this attention [9].

## 2.4 Parametric Models

Parametric models assume that entries of the covariance matrix follow a functional form, e.g., exponential or a power-law decay. This parametric function  $k : \chi^2 \rightarrow \mathbb{R}$  cannot be arbitrary; it has to be positive definite, i.e., it should satisfy

$$\int \int_{\chi^2} k(x, x') f(x) f(x') dx dx' \geq 0,$$

for all  $f \in L_2(\chi)$ , where  $\chi$  is the input space.

Gaussian Processes (GP) are a general framework for modeling random processes as realization from a jointly Gaussian model with a specified parametric covariance function [13]. While modeling random processes, the task of estimating the covariance function the adapted GP will have is equivalent to determining what that GP will exactly be. Therefore, this is a vital intrinsic task of the framework. To that end, [13] presents the following table of some possible positive definite functions as candidates for covariance function:

covariance function	expression
constant	$\sigma_0^2$
linear	$\sum_{d=1}^D \sigma_d^2 x_d x_d$
polynomial	$(x \cdot x' + \sigma_0^2)^p$
squared exponential	$\exp(-\frac{r^2}{2l^2})$
exponential	$\exp(-\frac{r}{l})$
$\gamma$ -exponential	$\exp(-(\frac{r}{l})^\gamma)$
rational quadratic	$(1 + \frac{r^2}{2\alpha l^2})^{-\alpha}$

**Table 2.1:** Summary of several commonly-used covariance functions.

Not only one is limited to these covariance functions, but also it is possible to create new covariance functions from existing ones. New covariance functions can be obtained by several separate operations varying from the simple ones such as plain summation and multiplication to more complicated ones such as convolution.

Both of the problems of choosing the appropriate covariance function and choosing the "hyperparameter"s of the corresponding covariance function are referred to as "model selection" or "training of the Gaussian process" and needed to be addressed in order to find a covariance function estimate at the end. Generally the family and the parameters of this covariance function estimate is sought to be selected such that average error with this estimate is minimized on unseen test examples. This covariance function estimate  $k$  can be used to find the covariance matrix estimate  $P$  through  $[P]_{(i,j)} = \text{Cov}(x_i, x_j) = k(x_i, x_j)$ . It should be noted that Gaussian processes are very flexible and allow to define the covariance over the

continuous domain, interpolating the covariance function in between the samples that we have seen. For our application this is typically not needed, as expirations of financial contracts occur at specified discrete times, and interpolation is not needed.

The major shortcoming of parametric models is again its strict parametric restriction. Although it presents the opportunity to model covariance function in many different forms, the true covariance matrix may not actually be following any of these forms.

## 2.5 Other methods

Besides the methods presented, which are among the most frequently used for covariance matrix estimation, there are other methods again with assumptions on the structure of the data. One of these approaches, which we may call *banded approximation model* [30], consists of allowing only a certain number of non-zero diagonals (bands) in, namely banding or tapering, the sample covariance matrix or its inverse, the latter achieved by the Cholesky decomposition of the inverse (the latter also has connections with graphical models). One downside of this approach is that the classes of covariance matrices that is described by [30] and referred to as the classes for which banding makes sense, such as

$$\mathbb{K}(m, C) = \{P : [P]_{(i,i)} \leq Ci^{-m}, \text{ for all } i\},$$

are very restrictive.

Another approach is to calculate the shrinkage estimates, which take a weighted combination of the sample covariance matrix and a strongly-regularized model (such as low-rank). For example in [31] this strongly-regularized model corresponds to the identity matrix. While the general approach does improve the expected mean-square error, it does not add any new kind of structure.



## Chapter 3

# Smooth and Monotone Formulation for Covariance Matrix Estimation

This chapter contains our formulation of covariance estimation as an optimization problem involving a data fidelity term as well as constraints imposing smoothness and monotonicity of the covariance matrix. We first motivate our formulation in Section 3.1. Following, in Section 3.2 we formulate the estimation problem in a convex-optimization framework, and propose solving the resulting semidefinite-programming problem by an interior-point method. In Section 3.3, we demonstrate the application of our approach on a number of examples with limited, missing and asynchronous data, and show that it has the potential to provide more accurate covariance matrix estimates than existing methods, and exhibits a desirable eigenvalue-spectrum correction effect.

■ **Bibliographic notes.** The formulation is originally of D. M. Malioutov, who presented first it in his short paper [28], from which therefore major part of Section 3.2 is borrowed, especially up until to Subsection 3.2.2. Some of the rest of this chapter is based on our joint submission [29] reflecting research done in collaboration with D. M. Malioutov, as parts of the rest of the thesis are.

### 3.1 Motivation

**W**E now introduce our setting for covariance regularization and motivate some relevant applications. Our starting assumption is that the random variables of interest have an ordering according to some manifold – for example they may be acoustic measurements at microphones along a linear or a spatial array, or they may be the changes in prices for interest rates along an interest rate curve. We aim to

estimate the spatial (cross-sectional) covariance matrix for these types of random variables from a scarce number of samples.

Parametric models of covariances may be too restrictive for some applications and may introduce strong bias. We instead consider a non-parametric approach which stipulates that the desired correlation structure “respects” the manifold ordering – namely – the covariance function is monotonic with the manifold distance and, furthermore, it is well-behaved – continuous or even smooth – along the manifold. Both of these are very natural assumptions when dealing with spatial data. For example in the sensor network case: if sensor  $i$  is located closer to sensor  $j$  than to sensor  $k$ , then we expect the correlation  $[P]_{(i,j)}$  to be higher than  $[P]_{(i,k)}$ . This corresponds to the monotone structure in the covariance matrix. Also, knowing the physics of the noise generating process one may have strong evidence not to expect discontinuities in the correlation structure. This second point enables us to impose smoothness for off-diagonal entries in the covariance matrix.

As implied in the first paragraph of this section, we can assume this kind of structure not only for the case of modeling data covariances in sensor arrays and other engineering topics such as computer vision, but also for computational finance problems such as interest rate modeling. In the interest-rate example, when the  $i$ -th contract is located closer to the  $j$ -th contract than to the  $k$ -th one, then we expect the correlation  $[P]_{(i,j)}$  to be higher than  $[P]_{(i,k)}$ . Also, again there is rarely a good reason to expect discontinuities in the correlation structure.

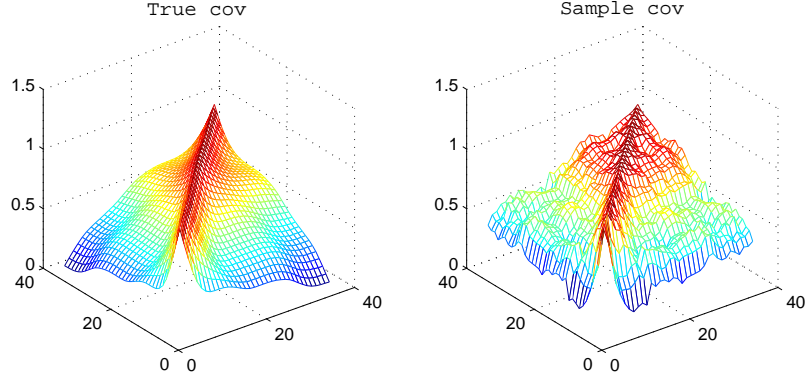
## 3.2 Formulation and Solution through IPM

### 3.2.1 Formulation

We now formulate the regularization problem for smooth isotonic covariances for processes consisting of variables that exhibit a natural ordering on a line. Suppose we have a zero-mean (without loss of generality) random vector  $\mathbf{x}(t)$ , where  $\mathbf{x}(t) = (x_1(t), \dots, x_N(t))^T$ . We are interested in the spatial covariance matrix of  $\mathbf{x}$ ,  $P^* = E[\mathbf{x}\mathbf{x}^T]$ , and also in the matrix of the correlation coefficients,  $[\tilde{P}^*]_{(i,j)} \propto \frac{[P^*]_{(i,j)}}{\sqrt{[P^*]_{(i,i)}[P^*]_{(j,j)}}}$ . We assume that temporal correlation is negligible and ignore it in this thesis. Suppose that only a small number of samples  $\mathbf{x}(t_1), \dots, \mathbf{x}(t_T)$  are available with  $T$  comparable or even smaller than  $N$ . We aim to leverage the assumptions of monotonicity and smoothness to get a better estimate of  $P^*$  than the ordinary sample covariance matrix  $\hat{P} \triangleq \frac{1}{T} \sum_i \mathbf{x}(t_i)\mathbf{x}(t_i)^T$ .

To that end we formulate a regularized covariance estimator. Let  $\mathcal{M}$  be the class





**Figure 3.1:** Term-rate covariances: (a) true (b) sample estimate.

of monotone positive semi-definite (psd) covariance functions:

$$\mathcal{M} = \{P | P \succeq 0, [P]_{(i,j)} \geq [P]_{(i,k)} \text{ for } i < j < k\}. \quad (3.1)$$

Then, we obtain our first monotonic estimate of the covariance as follows:

$$\min_P D(P, \hat{P}) \quad \text{such that} \quad P \in \mathcal{M} \quad (3.2)$$

where  $D(P, \hat{P})$  is an error metric of our choice: we will use the squared Frobenius norm

$$D(P, \hat{P}) = \|P - \hat{P}\|_F^2 = \sum_{i=1}^N \sum_{j=1}^N \left( [P]_{(i,j)} - [\hat{P}]_{(i,j)} \right)^2, \quad (3.3)$$

but KL-divergence and the operator norm are also possible. Note that the constraint set  $\mathcal{M}$  is a convex set, with linear and positive definiteness constraints, and for natural choices of the metric  $D$  the objective will also be convex. When  $D$  is either the operator norm or the Frobenius norm, our regularizer can be found as a solution to a semi-definite programming problem (SDP). For the error metric, we can also use Kullback-Leibler (KL) divergence, which for two-zero mean Gaussian distributions with covariances  $P$  and  $Q$  is defined as

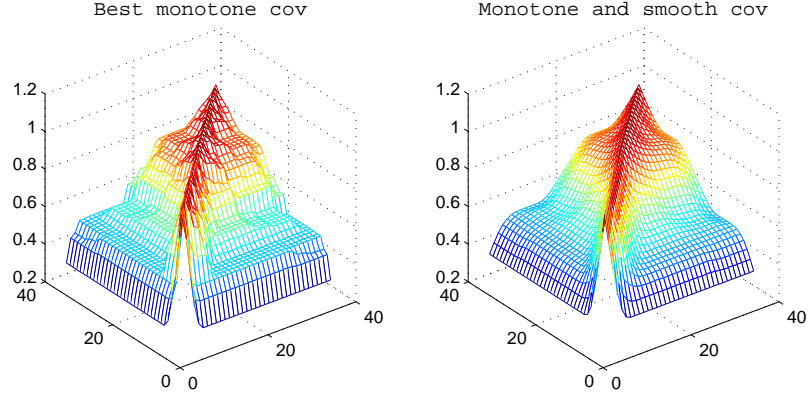
$$D(P||Q) = \frac{1}{2} [\log(\det Q P^{-1}) + \text{tr}(Q P^{-1}) - N] \quad (3.4)$$

**Remark.** If the true covariance indeed belongs to  $\mathcal{M}$ , then projecting the sample covariance onto  $\mathcal{M}$  is guaranteed to decrease the error<sup>1</sup> due to the contraction property of projections onto convex sets:  $\|\Pi_{\mathcal{M}}(\hat{P} - P)\| \leq \|\hat{P} - P\|$ .

We now consider a computational example that we describe in detail in Section 3.3. In Figure 3.1 we plot a true smooth-monotone covariance, and the sample estimate exhibiting finite-sample noise. In Figure 3.2 (a) we see that simply restricting

---

<sup>1</sup>If the projection is done with respect to the same metric with which we evaluate errors, e.g., the Frobenius norm.



**Figure 3.2:** Covariance regularization: (a) monotone (b) monotone and smooth.

the covariance functions to be monotone produces a “staircase”-like effect. We do not expect natural phenomena to exhibit such discontinuous effects, and we also require that the covariance functions have some degree of smoothness. To that end we penalize the curvature over the surface of the covariance function  $P(x_1, x_2)$ , namely we penalize:

$$\int \int_S (\nabla^2 P(x_1, x_2))^2 dx_1 dx_2 \quad (3.5)$$

where  $S = \{(x_1, x_2) \mid x_2 > x_1\}$ . For a covariance matrix  $P$  this penalty imposes smoothness in the upper-triangular part, avoiding smoothing over the diagonal. To implement this numerically, over a discrete grid, we use the discrete version of the Laplacian operator on the manifold at the point of interest  $v$  and then sum its square over all  $v$ , yielding the penalty as

$$\sum_v (\nabla_v^2 f)^2, \quad \text{where } \nabla_v^2 f = \sum_{u \in N(v)} (f(x_u) - f(x_v)) \quad (3.6)$$

Here,  $N(v)$  is the set of neighbors of point  $v$ : for covariance  $[P]_{(i,j)}$  the neighbors of vertex  $v = (i, j)$  can be set to  $(i \pm 1, j)$ , and  $(i, j \pm 1)$ . The optimization problem is now:

$$\begin{aligned} \min_P \quad & D(P, \hat{P}) + \lambda \sum_v (\nabla_v^2(P))^2 \\ \text{such that} \quad & P \in \mathcal{M} \end{aligned} \quad (3.7)$$

where the parameter  $\lambda$  trades off smoothness with data-fidelity, and should ideally be chosen automatically, e.g., via cross-validation. The problem is still convex: the objective is convex quadratic, and the constraint set is semi-definite, making the problem an SDP. To see the benefit of enforcing smoothness we contrast covariance estimates in Figure 3.2 (a) and (b) and we see that this produces much smoother, and also, as we will see in Section 3.3, more accurate estimates.

**Remark.** It is straightforward to add additional constraints, e.g.,  $[P]_{(i,i)} = 1$  to (3.7) when dealing with correlation coefficient matrices, or positivity of correlations. In such cases it is only needed to change the constraint set  $\mathcal{M}$  from the class of monotone covariance functions into a smaller set  $\mathcal{M}'$  which respects the additional constraints.

### 3.2.2 Solution through IPM

In this subsection, we state that the optimization problem in (3.7) can be solved as an SDP and show in what form it can be solved via IPM. As it was mentioned in the end of the previous subsection, (3.7) is not only convex but also can be represented as a semidefinite optimization problem [22]:

$$\begin{aligned} \min \quad & \|P - \hat{P}\|_F^2 + \lambda \|M_s \circ (D_s P)\|_F^2 \\ \text{such that} \quad & P \succeq 0, \quad M_{m,l} \circ (D_{m,l} P) + M_{m,u} \circ (D_{m,u} P) \geq 0 \end{aligned} \quad (3.8)$$

where the operations  $M_s \circ (D_s P)$  and  $M_{m,l} \circ (D_{m,l} P) + M_{m,u} \circ (D_{m,u} P)$  encode smoothness and monotonicity constraints, respectively. In other words, we will have to select the matrices  $M_s$ ,  $D_s$ ,  $M_{m,l}$ ,  $D_{m,l}$ ,  $M_{m,u}$  and  $D_{m,u}$  such that

$$\|M_s \circ (D_s P)\|_F^2 = \sum_v (\nabla_v^2(P))^2, \quad (3.9)$$

$$\{P \mid M_{m,l} \circ (D_{m,l} P) + M_{m,u} \circ (D_{m,u} P) \geq 0\} \equiv \mathcal{M}. \quad (3.10)$$

We now first specify how we select these matrices and then explain its reasons.

Here,  $D_s$  is a  $(N-2) \times N$  matrix of zeros except that  $[D_s]_{(i, [i \ i+1 \ i+2])} = (1, -2, 1)$ ;  $M_s$  is a matrix of ones except that  $[M_s]_{(i, i-1)} = 0$ ;  $D_{m,l}$  and  $D_{m,u}$  are  $(N-1) \times N$  matrices of zeros except that  $[D_{m,l}]_{(i, [i \ i+1])} = (1 \ -1)$  and  $[D_{m,l}]_{(i, [i \ i+1])} = (-1, 1)$ ; and  $M_{m,l}$  and  $M_{m,u}$  are  $(N-1) \times N$  matrices of ones except that  $[M_{m,l}]_{(i,j)} = 0$  for  $i < j$  and  $[M_{m,u}]_{(i,j)} = 0$  for  $i \geq j$ .

Let us explain why the matrices are chosen such. The matrices  $D_s$  and  $D_{m,l}$ ,  $D_{m,u}$  compute differences of entries of  $P$  column-wise with weighting adjusted for corresponding encoding. However, these matrices go over all of the entries of  $P$  columnwise, and some of the entries of the products should be discarded for proper encoding. To encode smoothness properly, i.e., to exclude the diagonals of  $P$  from smoothness constraints, the masking matrix  $M_s$  of ones and zeros is multiplied with  $D_s P$  elementwise (denoted by the operator  $\circ$ ) so that the elements of  $D_s P$  corresponding to smoothing over the diagonal of  $P$  are multiplied with zero while other elements of  $D_s P$  are multiplied with 1. Similarly, to encode monotonicity properly, the masking matrices  $M_{m,l}$  and  $M_{m,u}$  of ones and zeros are multiplied elementwise with  $D_{m,l} P$  and  $D_{m,u} P$ , respectively.  $D_{m,l}$  subtracts each row from the above row in

$P$ , hence  $M_{m,l}$  is selected such that it keeps the lower triangular part of the product  $D_{m,l}P$  in order to represent column-wise monotonicity in lower triangular part of  $P$ . Similarly  $M_{m,u}$  and  $D_{m,u}$  are selected such that  $M_{m,u} \circ (D_{m,u}P)$  represents column-wise monotonicity in upper triangular part of  $P$ . Since positive semi definiteness is another constraint on  $P$ , column-wise monotonicity automatically guarantees row-wise monotonicity in  $P$  as well.

The resulting SDP problem (3.8) can be readily solved via an interior point method using one of a number of standard SDP optimization packages, e.g., SDPT3 [23]. We are also using SDPT3 through YALMIP for the experiments of this chapter.

Note that again it is straightforward to add additional constraints, e.g.,  $[P]_{(i,i)} = 1$  or positivity of correlations to (3.8), as it was for (3.7). For instance, the mentioned constraints can be added by including  $M_d \circ P = M_d$  or  $P \geq 0$  in the constraint set of (3.8), where  $M_d$  is a diagonal  $N \times N$  matrix of ones and  $\geq$  is elementwise comparison. Such modifications don't change the fact that the problem is still an SDP and it can be readily solved through IPM.

## A Brief Glance at Interior Point Methods

We now present a brief look at IPMs, summarizing from [35]. Due to the existence of extensive theoretical results on their convergence and complexity combined with their extreme reliability in practice, the use of IPMs is most attractive not only for SDP but also for other special classes of convex programming such as second order cone (SOC) programming and linear programming (LP), while they can also be applied to various other linear and nonlinear programs.

Let us consider a general convex problem with a convex constraint region  $\chi$ . The basic idea of IPMs is to introduce a parameterized family of problems augmented with a barrier function for the convex region. The barrier function has to be well-defined in the feasible region, smooth, strongly convex<sup>2</sup> in the interior of  $\chi$ , and increase to infinity as the boundary of  $\chi$  is approached. By applying a barrier function, the convex constrained problem is transformed into a family of convex unconstrained problems which are infinite outside the feasible region of the original problem. Then, Newton's method is applied with reweighing on these problems to find a point sufficiently close to the original problem.

For the class of convex problems a theory of self-concordant<sup>3</sup> barriers has been developed [22], which can be used to prove polynomial-time convergence for all

---

<sup>2</sup> $f(x)$  is strongly convex if it is convex, and additionally,  $(\nabla f(x) - \nabla f(y))^T(x - y) \geq \alpha \|x - y\|_2^2$  for some  $\alpha > 0$ , and  $\forall x, y \in \mathbb{R}^n$

<sup>3</sup>Self-concordance means satisfying several properties with respect to the local variability of the Hessian of the barrier function.

convex problems, including IPMs, when self-concordant barriers are used (for more details refer to [1], and references therein). However, the worst-case results are not representative of the average-case performance which is seen in practice. It has been observed that using a faster step-size rule and a greater number of Newton’s steps leads to much better practical performance. The worst case complexity results for long-step IPMs however do not provide strong guarantees, unlike their short-step counterparts.

### 3.3 Experimental Results

In this section we apply our proposed covariance estimation approach on an important problem in mathematical finance. In particular, we consider the problem of term-structure modeling, which we first describe in the following subsection and then on which we apply our method through Subsections 3.3.2 - 3.3.4, going through mainly comparisons with other methods as well as a modification of our method for the special scenario of missing data. In Subsection 3.3.5, we take a different direction and show how our method can be valuable for consistency of the eigenvalue spectrum of the covariance matrix estimate in the high-dimensional setting with limited data.

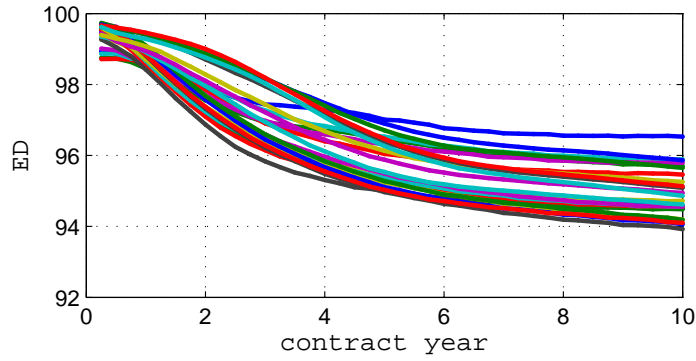
#### 3.3.1 Term-structure Modeling

We now describe the interest-rate risk modeling problem that we will use as an example of our smooth-monotone covariance estimation framework. The interest rate curve describes the available interest rate as a function of the duration for which the investment is locked in. The curve changes with time and takes on a variety of different shapes. We expect the correlations between variables in the curve to be monotonic with respect to the difference in duration, and also expect not to have persistent discontinuities in such a structure – thus fitting well with the framework in this thesis. (We note that the approach does not directly apply to equities data, as there is no natural manifold ordering.)

To be specific we will look at the Eurodollar (ED) curve, which describes the prices of the Eurodollar futures contract with expiration  $k$  quarters (multiples of 3 months) from the present. For historical reasons Eurodollars are measured as  $100 - x$  where  $x$  is the interest rate for the contract. Some sample Eurodollar futures curves as a function of time to expiration (delivery) are shown in Figure 3.3 for a few different dates, with linear interpolation in between contracts<sup>4</sup>.

---

<sup>4</sup>Data used with permission from the Wall Street Journal online.



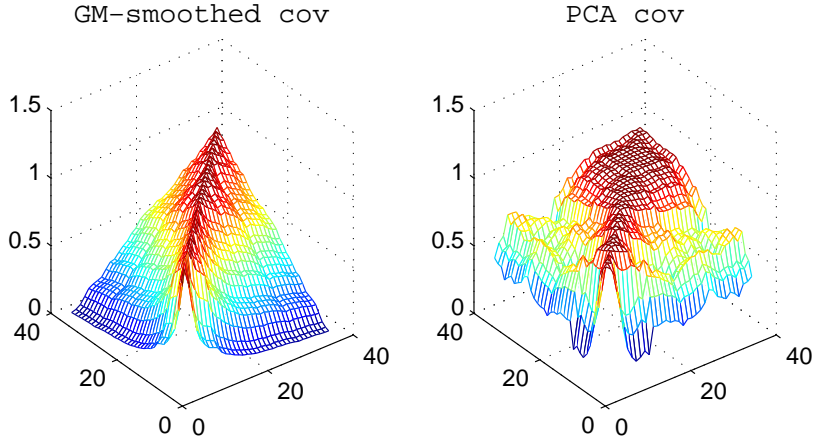
**Figure 3.3:** Sample ED curves, linearly interpolated.

At any time point  $t$ , we order the unexpired contracts with respect to their time left to expiration (beginning from the closest one to expire) and call this index  $i$  (the difference between the time left to expiration of two nearby contracts is 1 quarter, i.e., 3 months). Therefore, if we let the random variable  $y_i$  represent the price of the  $i$ -th Eurodollar contract, then  $y_i(t)$  is its realization at time  $t$ . When the 1<sup>st</sup> contract expires (e.g., on March 11<sup>th</sup>, 2013), all the contracts roll over, i.e., the 2<sup>nd</sup> contract (with expiration date June 11<sup>th</sup>, 2013) becomes the 1<sup>st</sup>, the 3<sup>rd</sup> contract (with expiration date September 11<sup>th</sup>, 2013) becomes the 2<sup>nd</sup>, ..., the  $(N+1)$ -th contract becomes the  $N$ -th, and so on.

Continuing with definitions, the  $i$ -th spread is defined as the random variable  $x_i \triangleq y_i - y_{i+1}$ , i.e., as the difference in nearby contracts, whose realization in each time point  $t$  is  $x_i(t) = y_i(t) - y_{i+1}(t)$ . What we are interested in is the risk for a portfolio of ED spreads, which will be defined via the covariances for daily changes in prices of the spreads, i.e., via the covariance matrix of the random variables  $\Delta x_i$ , whose each realization is  $\Delta x_i(t) = x_i(t) - x_i(t-1)$ , for  $i = 1, \dots, N$  (we only consider the first  $N$  spreads).

The reason we are focusing on the spreads instead of the future contracts is that the main component of risk in term-rate models is a parallel shift, i.e., same amount of change in the prices of all future contracts. One is almost hedged against (i.e., not sensitive to) parallel shifts in the curve if no net position is taken in future contracts. Therefore, practically, for some trading desks the requirement is that they are almost flat futures, but they can hold some spreads, for risk purposes.

A popular model for the term-rate curve is based on PCA and approximates the covariance by three main PCA factors, having informal interpretation of level, slope, and curvature. However, a more accurate covariance model may be desired when dealing with spreads, as the dominant first principal component gets largely removed. It should also be noted that for this reason using the Eurodollar futures curve



**Figure 3.4:** Alternative estimates: (a) MRF (b) PCA.

for our experiments and focusing on risk in ED spreads makes covariance estimation more challenging.

### 3.3.2 Experiments with a Known Underlying Covariance Matrix

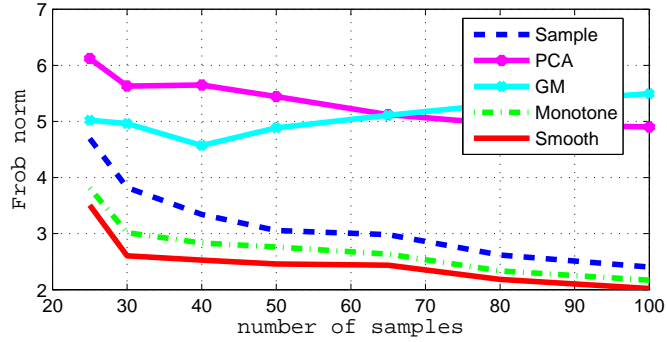
In our first experiment we generate a number of samples from a known smooth monotone covariance  $P$ , and use them to estimate  $P$ . It should be noted that in practice one never has the 'true' covariance – here we took a sample covariance from ED data, and smoothed it, as a proxy for the true one.

In Figure 3.1 we show the true covariance, and the sample estimate, for the case of  $N = 40$ ,  $T = 40$ . In Figure 3.2 we apply (a) our monotone and (b) smooth-monotone regularization<sup>5</sup> from (3.2) and (3.7), respectively. We can see that while the monotone version suffers from the stair-case effect, the smoothed version looks qualitatively close to the true covariance.

For comparison we compute the PCA estimate with  $K = 3$  and an MRF model estimate with the information matrix restricted to have  $K = 5$  non-zero diagonals, learned by iterative proportional fitting (IPF). Figure 3.4 shows that the estimated covariance matrices exhibit only a rough similarity to the original.

Finally, we compute the average Frobenius error over 25 trials and present the results in Figure 3.5 for all the methods, as a function of the number of available samples. The Gaussian MRF, and the PCA methods are biased for fixed  $K$ : the estimates do not improve with more samples. However, the monotone and the smooth-monotone estimates provide a significant improvement in accuracy over the

<sup>5</sup>For simplicity  $\lambda$  was set by trial and error in these experiments.



**Figure 3.5.** Errors of sample covariance, monotonic, smooth, GM and PCA estimates.

sample covariance. These results suggest the appeal of our approach.

### 3.3.3 Missing data

Missing data plagues all of applied science and has many incarnations in finance: (i) prices for illiquid instruments may not be available for days, (ii) related products in the futures space may have different expirations: one product may expire while a related product is still active, (iii) holiday schedules in different countries do not generally agree: prices for financial instruments are not available when the corresponding market is closed. To illustrate robustness to missing data we consider an example where some entries in the sample covariance matrix are missing (unknown).

Suppose that we have  $\hat{P}$  for only some subset  $\mathcal{I}$  of entries:  $(i, j) \in \mathcal{I} \subset \{1, \dots, N\}^2$ , and no observations for the rest. Our smooth-monotone optimization formulation can be immediately adapted to this setting:

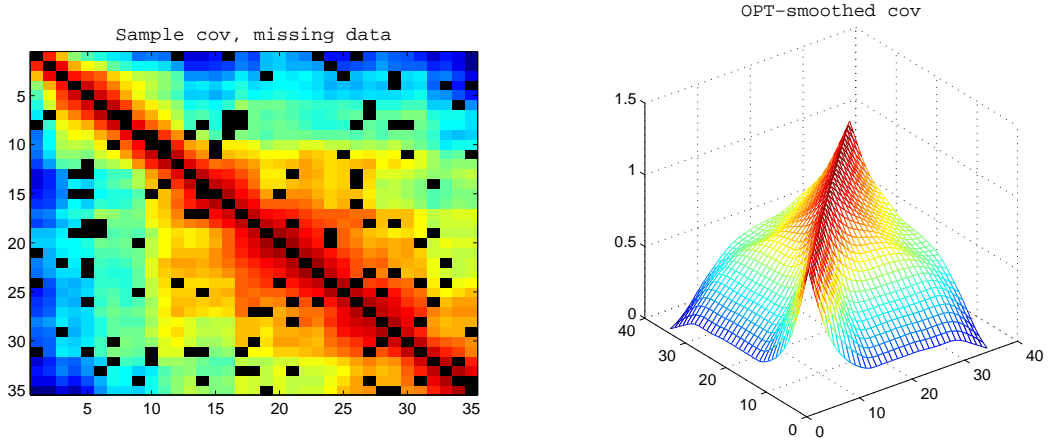
$$\begin{aligned} \min_P \quad & D_{\mathcal{I}}(P, \hat{P}) + \lambda \sum_v \nabla_v^2(P) \\ \text{such that} \quad & P \in \mathcal{M}, \end{aligned}$$

where, since for  $D$  we use the squared Frobenius norm,  $D_{\mathcal{I}}$  becomes

$$D_{\mathcal{I}}(P, \hat{P}) = \sum_{(i,j) \in \mathcal{I}} \left( [P]_{(i,j)} - [\hat{P}]_{(i,j)} \right)^2$$

This does not affect the convexity of the problem, and can be solved using the same optimization methods. We continue our interest-rate curve example, with  $N = 40$ , and  $T = 50$  samples, and 10-percent of the samples missing. The sample covariance matrix for this case is shown in Figure 3.6 (a). For this missing data problem, the covariance matrix estimated by our proposed approach is shown in 3.6 (b). Clearly the approach is very robust against moderate missing data and recovers a very similar estimate to the fully observed case.





**Figure 3.6.** (a) Sample covariance with missing data. (b) Recovered smooth-monotone covariance.

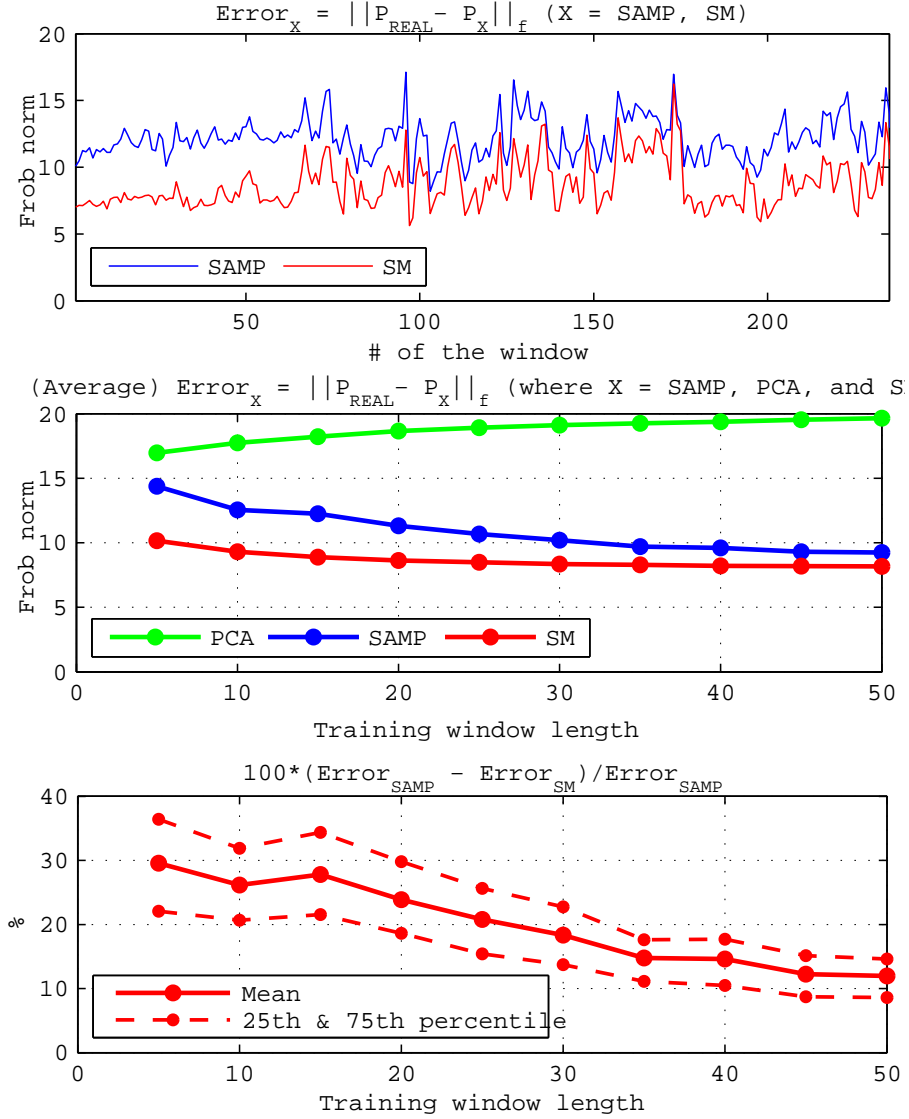
### 3.3.4 Out-of-sample Covariance Prediction

We now present a study of forecasting future correlation coefficient matrices over several years of historical data of ED prices. The accuracy of this prediction is crucial since portfolio selection methods, such as Markowitz portfolios, depend on it in order to optimally allocate assets.

To be specific, we first compute the sample correlation coefficient matrix over a training window of  $T_{TR}$  business days and use this matrix to estimate the correlation coefficient matrix using our proposed method as well as alternative methods. We then compute the realized (i.e., sample) correlation coefficient matrix over a test window, of  $T_{TEST}$  business days immediately following the training window, and compare it to our forecasts. The experiment uses running windows with shifts of 5 business days over the course of a year. The data set we use contains information regarding 40 future contracts at a time, corresponding to 39 spreads; therefore, for correlation coefficient matrix sizes of 40 or larger we artificially create new future contracts by linearly interpolating the daily prices of already existing futures and calculate new spreads from these new future contracts.

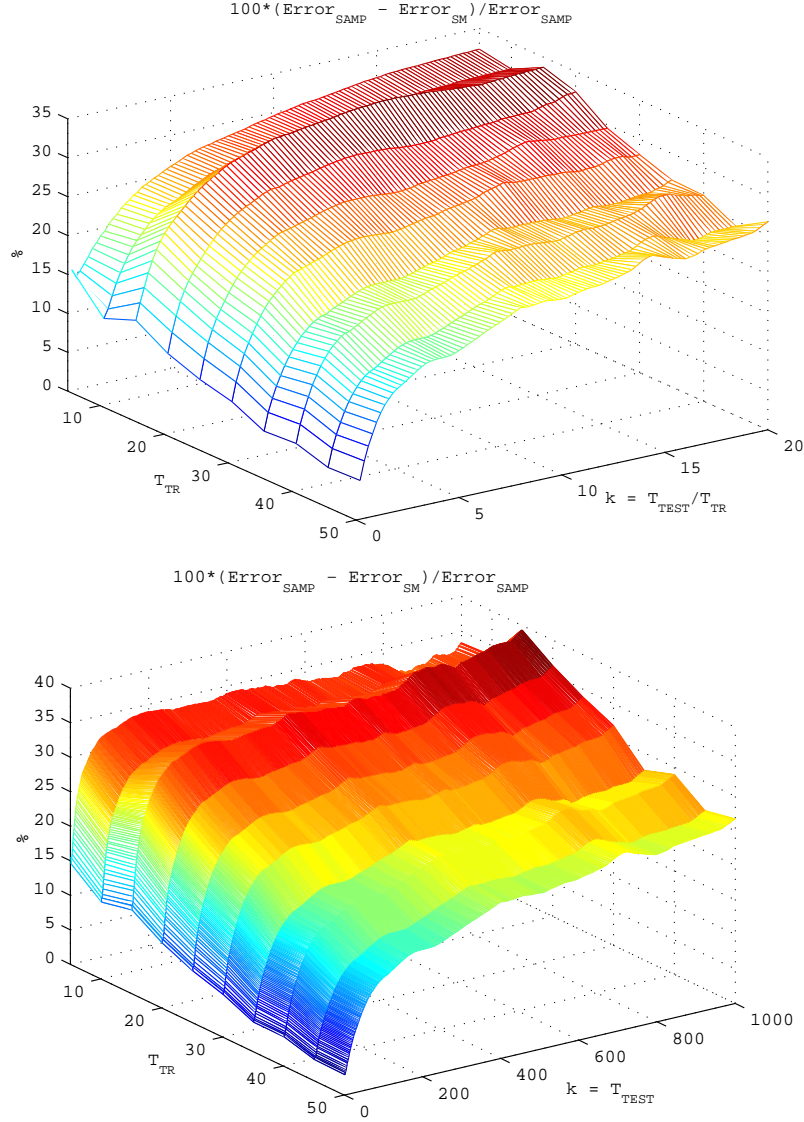
In Figure 3.7 we show the average error (in Frobenius norm) as a function of  $T_{TR}$  with  $T_{TEST}$  set to 50, when the ratio  $k \triangleq T_{TEST}/T_{TR}$  is 4. We observe that  $P_{SM}$  performs significantly better and gives much smaller errors than the other methods. Figure 3.7 (c) shows the percentage improvement of smooth-monotone over the sample estimate. Smooth and monotone regularization appears especially valuable for small  $T_{TR}$ , demonstrating its robustness in forecasting risk in scenarios with severely limited data.

Since Figure 3.7 (c) shows the performance of  $P_{SM}$  versus  $\hat{P}$  for a single value



**Figure 3.7.** (a) Frobenius error over running windows. (b) Average Frobenius errors as a function of training window length. (c) Average percentage improvement of forecast error from  $P_{SM}$  over  $\hat{P}$ .

of  $k$ , we extend this analysis of percentage improvement of smooth-monotone over the sample estimate to several  $k$  in Figure 3.8 (a). Here we can see that the relative performance of  $P_{SM}$  improves as  $k$  increases and observe the same relative behavior over  $T_{TR}$  for all  $k$ . As an example for the latter,  $P_{SM}$  outperforms  $\hat{P}$  most always when  $T_{TR}$  is around 15. The improvement  $P_{SM}$  provides over  $\hat{P}$  becomes more pronounced for relatively smaller training windows, demonstrating its robustness in limited data scenarios. To provide further evidence, in Figure 3.8 (b) we present a similar analysis, this time for several  $T_{TEST}$  values instead of  $k$ , and observe a similar behavior.

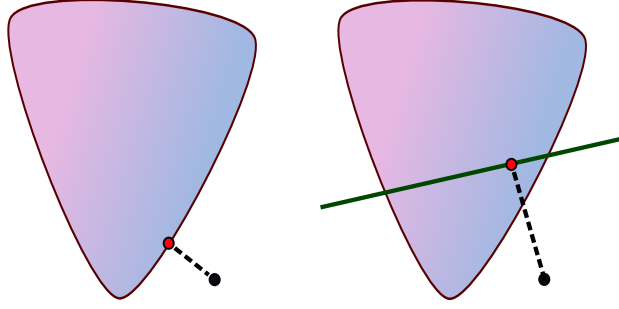


**Figure 3.8.** What percent  $P_{SM}$  outperforms  $\hat{P}$  (on average) (a) for  $T_{TR}$  vs  $k \triangleq T_{TEST}/T_{TR}$  (b) for  $T_{TR}$  vs  $T_{TEST}$ .

### 3.3.5 Spectral Correction

As we mentioned in Chapter 2, in Section 2.1 to be specific, one of the symptoms of a catastrophic breakdown of the sample covariance estimate in the high-dimensional setting with limited data is the inconsistency of the eigenvalue spectrum. We provide motivation and conduct a numerical study showing that smooth and monotone regularization can help dramatically in that respect.

A particularly important and challenging case for correcting the eigenvalue spectrum of a sample covariance matrix is the *asynchronous setting* arising in practical applications. When several time series occur at different temporal resolutions, in order to estimate the covariance matrix it is easiest to look at each pair of time se-



**Figure 3.9:** Projection onto the p.s.d. cone vs. smooth and monotone solution.

ries, align<sup>6</sup> them, and compute the pairwise covariance. However, once this is done for each pair, the covariance matrix is no longer guaranteed to be positive definite. Previous solutions to fix this defect simply projected the covariance matrix onto the space of p.d. matrices [27]:

$$\min D(P, \hat{P}) \quad \text{such that} \quad P \succ 0 \quad (3.11)$$

For the case of  $D(P, \hat{P}) = \|P - \hat{P}\|_F^2$ , there is a closed form solution based on the eigen-decomposition: take  $\hat{P} = USV^T$ , where  $U$  and  $V$  are orthogonal, and  $S$  is diagonal. Then the solution to (3.11) is simply

$$P^* = U \max(S, 0) V^T, \quad (3.12)$$

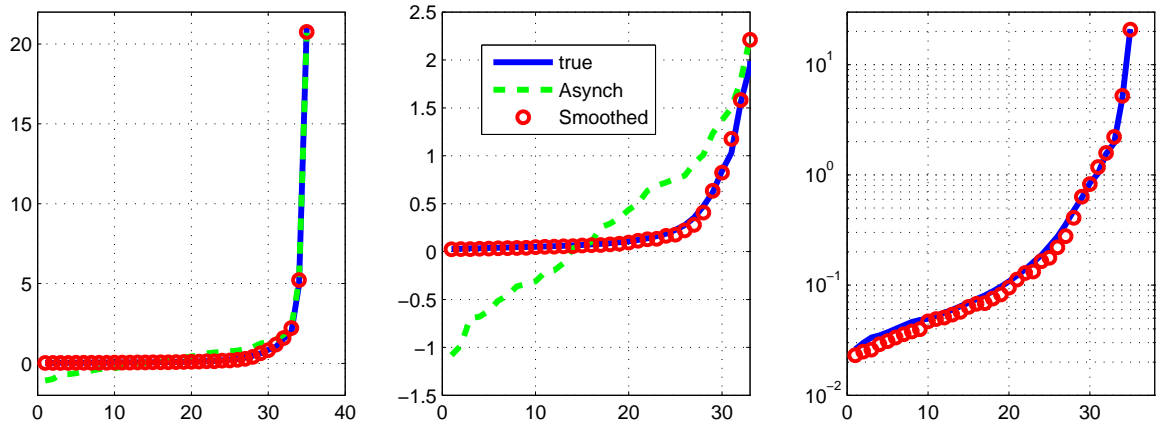
i.e., the negative eigenvalues are set to zero.

We consider the merits of our smooth-monotone approach in spectral correction. The motivation (mainly to build intuition, and not to be taken too literally) is illustrated in Figure 3.9. When we find the closest p.s.d. matrix, we simply project  $\hat{P}$  onto the p.s.d. cone. However when we have side-information of smoothness and monotonicity, then our approach will tend to guide the solution into the interior of the p.s.d. cone and closer to the correct solution.

We consider a numerical example with  $N = 36$ , and  $T = 40$  asynchronous samples: each  $P_{ij}$  is estimated from a pairwise sample  $\{x_i(t), x_j(t)\}_{t \in 1, \dots, T}$ , drawn independent of other pairs. We fix  $P$  to be unit-diagonal. We use the asynchronous covariance as  $\hat{P}$ , and apply our approach in (3.7). In Figure 3.10 we plot eigenvalues of the (i) true covariance matrix, (ii) asynchronous sample covariance (iii) smooth-monotone fit to the sample covariance. The left plot shows the spectra, with the detail shown in the middle plot. Sample-covariance spectrum breaks down completely, with about half of the eigenvalues negative. Projection onto the p.s.d.

---

<sup>6</sup>For example one can re-sample the time series to include time points from both. This would be very costly for much more than two series together.



**Figure 3.10.** (a) True, sample, and smooth-monotone eigen-spectra. (b) detail  
(c) log-scale of true and smooth-monotone spectra

cone would simply set the negative eigenvalues to zero, leaving the positive mis-estimated eigenvalues intact. However, the smooth-monotone eigenvalue spectrum follows the true one closely. A log-plot of the true and smooth-monotone spectra appears in the right-most plot, and we see that indeed we match the spectrum very closely! This experiment suggests that smooth-monotone regularization can be very effective in spectral correction for covariance estimation, and is especially valuable for asynchronous settings.



## Chapter 4

# Fast Algorithms for Smooth and Monotone Covariance Matrix Estimation

**R**ECALL that in Chapter 3 we presented the smooth and monotone optimization problem in (3.7) as:

$$\begin{aligned} \min_P \quad & D(P, \hat{P}) + \lambda \sum_v (\nabla_v^2(P))^2 \\ \text{such that} \quad & P \in \mathcal{M} \end{aligned}$$

and solved the resulting semidefinite optimization problem given below (and in (3.8)) via an IPM:

$$\begin{aligned} \min \quad & \frac{1}{2} \|P - \hat{P}\|_F^2 + \lambda \|M_s \circ (D_s P)\|_F^2 \\ \text{such that} \quad & P \succeq 0, \quad M_{m,l} \circ (D_{m,l} P) + M_{m,u} \circ (D_{m,u} P) \geq 0 \end{aligned}$$

Solving SDP via an IPM can become unduly computationally expensive for large covariance matrices, as it involves computing the Hessian. In this chapter, we present an alternate perspective and develop optimal first-order methods for solving this optimization problem. Such methods are an exciting recent development in optimization, generalizing classical gradient projection by a clever use of smoothing and acceleration techniques [24, 25]. An important requirement to use such methods is that the projection onto the constraint set can be done efficiently. This can be achieved by considering the dual of the problem in (3.8).

Our ultimate aim in this chapter is to convey the principles of how we develop highly efficient first-order solvers for smooth and monotone covariance matrix estimation and to present this development. For pedagogical reasons, we first develop these ideas for the special case of our problem which contains monotonicity constraints only (see (3.2)). Therefore, we start with describing a dual first-order

method based on gradient projection [26] for our monotone problem in Section 4.2. Following, in Section 4.3, we develop a dual projected coordinate descent solution for our smooth and monotone problem, which is also a first-order method, inspired from the method developed for the monotone problem. In Section 4.4 we develop even faster versions first for our monotone problem and then for our smooth and monotone problem using FISTA, i.e., the optimal first order ideas of [25]. Finally, in Section 4.5 we present a detailed experimental analysis demonstrating the computational benefits offered by the algorithm we develop in this chapter.

However, first of all we start with revisiting the original gradient projection method developed by Boyd and Xiao [26] for a quick recall. This material will be useful in derivation of our algorithms in the subsequent sections.

## 4.1 Original Gradient Projection Method Revisited

The optimization problem Boyd and Xiao solve in [26], i.e., the *least-squares covariance adjustment problem* (LSCAP) is:

$$\begin{aligned} \min \quad & \frac{1}{2} \|X - G\|_F^2 \\ \text{such that} \quad & X \succeq 0, \\ & \mathbf{Tr} \, A_i X = b_i, \quad i = 1, \dots, p, \\ & \mathbf{Tr} \, C_j X \leq d_j, \quad j = 1, \dots, m, \end{aligned} \tag{4.1}$$

where the matrices  $X$ ,  $G$ ,  $A_i$ , and  $C_j$  are  $N \times N$  symmetric and real matrices. All of these matrices except  $X$  are given. The Lagrangian of LSCAP (4.1) can be simplified to:

$$L(X, Z, \nu, \mu) = \frac{1}{2} \|X - G\|_F^2 + \mathbf{Tr} \, X(-Z + A(\nu) + C(\mu)) - \nu^T b - \mu^T d, \tag{4.2}$$

where  $A(\nu) = \sum_{i=1}^p \nu_i A_i$ ,  $C(\mu) = \sum_{j=1}^m \mu_j C_j$ , and  $\nu_1, \dots, \nu_p$ ,  $\mu_1, \dots, \mu_m$ , and  $Z$  are Lagrange multipliers. Setting the gradient of  $L$  with respect to  $X$  to zero, the  $X$  that minimizes  $L$  is

$$X_m(Z, \nu, \mu) \triangleq G - A(\nu) - C(\mu) + Z. \tag{4.3}$$

Substituting this expression for  $X$  back into the Lagrangian  $L$  and obtaining the simplified dual function  $g(Z, \nu, \mu) = \inf_X L(X, Z, \nu, \mu) = L(X_m, Z, \nu, \mu)$ , the dual problem is found:

$$\begin{aligned} \text{maximize} \quad & g(Z, \nu, \mu) \quad \text{such that} \quad Z \succeq 0, \quad \mu \geq 0, \\ \text{where} \quad & g(Z, \nu, \mu) = -\frac{1}{2} \|G - A(\nu) - C(\mu) + Z\|_F^2 + \frac{1}{2} \|G\|_F^2 - \nu^T b - \mu^T d, \end{aligned} \tag{4.4}$$



Weak duality always holds for this problem [26]. Therefore, if the LSCAP (4.1) is *strictly feasible*, i.e., there exists an  $X \succeq 0$  that satisfies the constraints in (4.1), then *strong duality* holds: there exist  $Z^*$ ,  $\nu^*$ , and  $\mu^*$  that are optimal for the dual problem (4.4) and that give the optimal solution  $X^*$  of the LSCAP (4.1) via (4.3).

One of the critical points in [26] is that it is possible to analytically maximize  $g$  over the variable  $Z$  within the constraint  $Z \succeq 0$  by choosing

$$Z^* = (G - A(\nu) - C(\mu))_-. \quad (4.5)$$

Using this  $Z$  simplifies the dual problem (4.4) to

$$\begin{aligned} \text{maximize } & \psi(\nu, \mu) \quad \text{such that } \mu \geq 0, \\ \text{where } & \psi(\nu, \mu) = -\frac{1}{2}\|(G - A(\nu) - C(\mu))_+\|_F^2 + \frac{1}{2}\|G\|_F^2 - \nu^T b - \mu^T d. \end{aligned} \quad (4.6)$$

Defining  $X(\nu, \mu) \triangleq (G - A(\nu) - C(\mu))_+$ , the gradients of the dual objective are

$$\frac{\partial \psi}{\partial \mu_j} = \mathbf{Tr} \ C_j X(\nu, \mu) - d_j, \quad \frac{\partial \psi}{\partial \nu_i} = \mathbf{Tr} \ A_i X(\nu, \mu) - b_i. \quad (4.7)$$

After these steps, the dual projected gradient algorithm Boyd and Xiao suggest is

1. *Update  $X$ .* Set  $X := (G - A(\nu) - C(\mu) + Z)_+$ .
2. *Projected gradient update for  $\mu$  and  $\nu$ :*
  - (a) Evaluate dual gradients:  $\frac{\partial \psi}{\partial \mu_j} = \mathbf{Tr} C_j X(\nu, \mu) - d_j$ ,  $\frac{\partial \psi}{\partial \nu_i} = \mathbf{Tr} A_i X(\nu, \mu) - b_i$ .
  - (b) Set  $\mu_j := (\mu_j + \alpha \frac{\partial \psi}{\partial \mu_j})_+$ ,  $\nu_i := (\nu_i + \alpha \frac{\partial \psi}{\partial \nu_i})$ ,

The gradient has a Lipschitz constant of  $L = \sum_{i=1}^p \|A_i\|^2 + \sum_{j=1}^m \|C_j\|^2$ , and step size of  $\alpha < 2/L$  guarantees convergence.

## 4.2 Dual Projected Gradient Solution for the Monotone Problem

Recall that the monotone version of our problem was described in (3.2) as:

$$\min_P D(P, \hat{P}) \quad \text{such that} \quad P \in \mathcal{M},$$

which can be solved via IPM when it is presented in the following form:

$$\begin{aligned} \min \quad & \|P - \hat{P}\|_F^2 \\ \text{such that} \quad & P \succeq 0, \quad M_{m,l} \circ (D_{m,l} P) + M_{m,u} \circ (D_{m,u} P) \geq 0 \end{aligned} \quad (4.8)$$

which is in the same form as the smooth and monotone problem (3.8), except for the absence of the second (smoothing) term which exists in the objective of (3.8).

We can express our monotone problem in (4.8) fully in LSCAP form, enabling us to directly apply the gradient projection method to this LSCAP form. The objectives of (4.8) and (4.1) are the same, as the corresponding matrices from (4.8) are  $X = P$  and  $G = \hat{P}$ . There is no equality constraint in (4.8), so there will be no  $A_i$  in the corresponding LSCAP form. Therefore, for the transformation, the only thing we need to do is to transform the monotonicity constraint  $M_{m,l} \circ (D_{m,l}P) + M_{m,u} \circ (D_{m,u}P) \geq 0$  into the form of  $\mathbf{Tr}C_jP \leq d_j$  for  $j = 1, \dots, m$ .

The monotonicity constraint in (4.8) ensures that the elements of  $P$  in every column are in non-increasing order in the direction away from the diagonal, and the symmetry property of  $P$  due to the positive semi-definiteness constraint  $P \succeq 0$  extends this constraint to rows  $P$ . We can encode a non-decreasing constraint for each pair of column-wise neighbor elements of  $P$  as  $\mathbf{Tr}D_kP \geq 0$ , where  $\mathbf{Tr}D_kP$  calculates the difference  $[P]_{(i,j)} - [P]_{(i+1,j)}$  if  $i \geq j$  and  $[P]_{(i+1,j)} - [P]_{(i,j)}$  if  $i < j$  and where  $i$  and  $j$  are respectively the quotient and the remainder from the division of  $k$  by  $N$ , i.e.,  $i, j \in \mathbb{Z}_+$  are such that  $k = N(i-1) + j$ . To yield this calculation,  $D_k$  is selected as a  $N \times N$  matrix of zeros except that  $[D_k]_{([j-1]N+1),i} = (1, -1)$  if  $i \geq j$  and  $[D_k]_{([j-1]N+1),i} = (-1, 1)$  if  $i < j$ . Since there are a total of  $N \times (N-1)$  column-wise pairs of elements in  $P$ ,  $k$  runs from 1 to  $m = N(N-1)$ . The  $C_k$ 's we need to construct, however, are required to be symmetric, hence we assign

$$C_k = -(D_k + D_k^T), \quad k = 1, \dots, N(N-1), \quad (4.9)$$

where the added  $D_k^T$  encodes additionally non-decreasing constraint for the pair of row-wise neighbor elements that is symmetric with the column-wise pair in  $P$  (whose monotonicity is encoded with  $D_k$ ). This modification is still in accordance with the constraints in our monotone problem in (4.8) since the symmetry from the positive semi-definiteness constraint  $P \succeq 0$  in (4.8) extends this constraint to the rows  $P$  as mentioned above. The minus sign in (4.9) is to convert  $\mathbf{Tr}D_kP \geq 0$  to  $\mathbf{Tr}C_jP \leq d_j$ , and it is now clear that  $d_j$  should be selected zero for all  $j$ . To be in the same notation with [26] we will index  $C_k$ 's with  $j$  instead of  $k$ , and hence will be using the notation with  $C_j$ 's.

We can now express our monotone problem in the following LSCAP (primal) form, i.e., *monotone LSCAP*:

$$\begin{aligned} \min \quad & \frac{1}{2} \|P - \hat{P}\|_F^2 \\ \text{such that} \quad & P \succeq 0, \\ & \mathbf{Tr} C_j P \leq 0, \quad j = 1, \dots, N(N-1), \end{aligned} \quad (4.10)$$

---

**Algorithm 1** Dual Projected Gradient Solution for the Monotone Problem

---

**Init:** Set  $\mu = 0$ , pick step-size parameter  $\alpha$ .

**repeat**

    Compute  $C(\mu) = \sum_{j=1}^{N(N-1)} \mu_j C_j$

    Set  $P_k = (\hat{P} - C(\mu))_+$

    Compute gradients:  $\partial\psi/\partial\mu_j = \text{trace}(C_j P_k)$

    Let  $\mu_j = (\mu_j + \alpha \text{trace}(C_j P_k))_+$ .

**until** convergence

---

to which we will also refer as the *monotone problem* where it's appropriate.

The solution of this problem with its dual follows exactly the same path described in the previous section, just with no  $A_i$ 's and  $\nu_i$ 's involved. The key to be able to use this solution method is that strict duality holds for the monotone LSCAP (4.10) since it is strictly feasible.

**Claim.** Monotone LSCAP (4.10) is strictly feasible, i.e., there exists a  $P \succ 0$  that satisfies the linear inequalities in (4.10).

**Proof.** Let  $P = I \succ 0$ .  $I$  also satisfies the monotonicity constraints in (4.10). The latter can also be verified from the fact that  $\text{Tr} C_j P = \text{Tr} C_j I = \text{Tr} C_j \leq 0$  for all  $j = 1, \dots, m$  since for all  $j$  the diagonal of  $C_j$  by definition consists of integers 0 and sometimes one entry of  $-2$ . ■

The primal solution of our monotone LSCAP (4.10) is  $P^* = (\hat{P} - C(\mu^*))_+$ , where  $\mu^*$  is the optimal Lagrangian multiplier for the dual of the monotone LSCAP. To find this  $P^*$  via  $\mu^*$  as in gradient projection method, we implement the dual projected gradient algorithm as in Algorithm 1.

The Lipschitz constant for gradient  $\nabla\psi$  of the simplified dual objective  $\psi$ , i.e., the mapping from  $\mu$  to  $\partial\psi/\partial\mu$  determines the step size  $\alpha$  for which the convergence is guaranteed by the relation  $\alpha < 2/L$ . Since there is no  $A_i$  in our monotone LSCAP (4.10), using the value of  $L$  derived in [26] and mentioned in Section 4.1 gives us  $L = \sum_{j=1}^m \|C_j\|^2$ . There are one entry of  $-2$  and two entries of  $-1$  in  $C_j$ 's encoding monotonicity for the pairs of  $P$  whose one element is on the diagonal of  $P$ , while all of other  $C_j$ 's consist of two entries of  $1$ 's and two entries of  $-1$ 's (the rest of  $C_j$ 's are filled with zeros). Since there are  $2(N-1)$  of the first kind of  $C_j$ 's, there are  $m - 2(N-1) = N(N-1) - 2(N-1) = (N-1)(N-2)$  of the second kind of  $C_j$ 's. This yields

$$L = \sum_{j=1}^m \|C_j\|^2 = \sum_{j_1=1}^{2(N-1)} [(-2)^2 + 2(1^2)] + \sum_{j_2=1}^{(N-1)(N-2)} [2(1^2) + 2(-1)^2] = 4(N+1)(N-1). \quad (4.11)$$

Therefore a step size of  $\alpha < 1/2(N+1)(N-1)$  guarantees convergence. However,

the size of  $L$  which is  $O(N^2)$  makes this step size too small in practice, as this leads to too many iterations, increasing the computation time. In practical situations one can instead choose much more aggressive step sizes and still usually achieve convergence. We will refer to this fact and to value of  $L$  when we describe our experiments in Section 4.5.

### ■ Modification with additional constraints

We stated in Chapter 3 that it is straightforward to add additional constraints, e.g.,  $[P]_{(i,i)} = 1$  to our formulation when dealing with correlation coefficient matrices, or positivity of correlations. It is also straightforward to add such constraints to monotone LSCAP (4.10) and modify the solution algorithm accordingly. To provide an example, we now assume that we want to add the constraint  $[P]_{(i,i)} = 1$ .

In the modified monotone LSCAP, the constraint of  $[P]_{(i,i)} = 1$  takes form as  $\text{Tr} A_i P = 1$  for  $i = 1, \dots, N$ , where  $A_i$ 's are  $N \times N$  matrices of zeros except that  $[A_i]_{(i,i)} = 1$ . Hence the *modified monotone LSCAP* becomes

$$\begin{aligned} \min \quad & \frac{1}{2} \|P - \hat{P}\|_F^2 \\ \text{such that} \quad & P \succeq 0, \\ & \text{Tr } A_i P = 1, \quad i = 1, \dots, N, \\ & \text{Tr } C_j P \leq 0, \quad j = 1, \dots, N(N-1), \end{aligned} \tag{4.12}$$

to which we will also refer as the *modified monotone problem* where it's appropriate or simply as the *monotone problem with additional constraints*.

Again the solution of this problem with its dual follows exactly the same path described in previous section, this time including  $A_i$ 's and  $\nu_i$ 's. We are still able to use this solution method since the modified monotone LSCAP (4.12) is also strictly feasible: Again let  $P = I$ , which satisfies all of the constraints in (4.12).

The primal solution of our modified monotone LSCAP (4.12) is now  $P^* = (\hat{P} - C(\mu^*) - A(\nu^*))_+$ , where  $\nu^*$  comes into play as the additional optimal Lagrangian multiplier for the dual of the monotone LSCAP. To find this  $P^*$  via  $\mu^*$  and  $\nu^*$  as again in gradient projection method, we implement in Algorithm 2 a slightly modified version of the dual projected gradient algorithm.

As the final step, let us again calculate the Lipschitz constant  $L$  for gradient  $\nabla\psi$ . This time the corresponding mapping is from  $(\mu, \nu)$  to  $(\partial\psi/\partial\mu, \partial\psi/\partial\nu)$ , and  $L$  is given by  $L = \sum_{j=1}^{N(N-1)} \|C_j\|^2 + \sum_{i=1}^N \|A_i\|^2$ . We already know that the first sum term is  $4(N+1)(N-1)$ . Since each  $A_i$  we constructed consists of only one entry of 1 and the rest with zeros, the second sum term is  $N$ . This yields

$$L = 4(N+1)(N-1) + N, \tag{4.13}$$

without causing a dramatic change from the unmodified monotone LSCAP (4.10).

---

**Algorithm 2** Dual Projected Gradient Solution for the Monotone Problem with Additional Constraints

---

**Init:** Set  $\mu = \nu = 0$ , pick step-size parameter  $\alpha$ .

**repeat**

    Compute  $C(\mu) = \sum_{j=1}^{N(N-1)} \mu_j C_j$  and  $A(\nu) = \sum_{i=1}^N \nu_i A_i$

    Set  $P_k = (\hat{P} - C(\mu) - A(\nu))_+$

    Compute gradients:  $\partial\psi/\partial\mu_j = \text{trace}(C_j P_k)$  and  $\partial\psi/\partial\nu_i = \text{trace}(A_i P_k)$

    Let  $\mu_j = (\mu_j + \alpha \text{trace}(C_j P_k))_+$  and  $\nu_i = \nu_i + \alpha \text{trace}(A_i P_k)$ .

**until** convergence

---

### 4.3 Dual Projected Coordinate Descent Solution for the Smooth and Monotone Problem

Now we turn back to our smooth and monotone problem (3.8):

$$\begin{aligned} \min \quad & \frac{1}{2} \|P - \hat{P}\|_F^2 + \lambda \|M_s \circ (D_s P)\|_F^2 \\ \text{such that} \quad & P \succeq 0, \quad M_{m,l} \circ (D_{m,l} P) + M_{m,u} \circ (D_{m,u} P) \geq 0 \end{aligned}$$

The constraints of this problem are the same as those of our monotone problem (4.8) which was transformed into the LSCAP form (4.10). Following a similar development, we can express our smooth and monotone problem in the following form:

$$\begin{aligned} \min \quad & \frac{1}{2} \|P - \hat{P}\|_F^2 + \lambda \|M_s \circ (D_s P)\|_F^2 \\ \text{such that} \quad & P \succeq 0, \\ & \text{Tr } C_j P \leq 0, \quad j = 1, \dots, N(N-1). \end{aligned} \tag{4.14}$$

However, this form is not fully in LSCAP form (4.1) due to the extra second (smoothing) term in the objective. Although it may look like that this extra term wouldn't change the solution of the dual problem much, it actually substantially complicates it. First of all, the gradients become more complicated due to this smoothing term. However, most important of all, the smoothing term prevents us from deriving a closed form  $Z^*$ , which is the Lagrangian multiplier  $Z$  that is optimal for the dual of the problem (4.14) over the choice of  $Z \succeq 0$ . We will show this explicitly in Subsection 4.3.2. In Section 4.1 we emphasized that while solving the original LSCAP (4.1) we are able to find this  $Z^*$ , which analytically maximizes the dual problem (4.4) over  $Z \succeq 0$ . This is crucial since we cannot find the optimal  $Z^*$  by using regular matrix calculus, i.e., by taking derivatives, just like that it is not possible to find optimal values  $\mu^*, \nu^*$  of the other Lagrangian multiplier in this way. Therefore, the only way to find a closed form of  $Z^*$  is analytical, and this is

not possible for the dual of our problem (4.14) since, as we will show in Subsection 4.3.2,  $Z^*$  appears inside a number of terms in the dual function instead of just one term in the dual function (4.4) of the original LSCAP.

The fact that we cannot find a closed form  $Z^*$  enforces us to do descent in  $Z$ , just like we already do in  $\mu$  and  $\nu$ . We call the solution method we derive this way in Subsection 4.3.2 *(dual) projected coordinate descent solution for the smooth and monotone problem*. However, since it is simpler and sets a good introductory example, let us first show in the following Subsection 4.2.1 the application of this solution for the LSCAP (4.1), namely, how we would solve the LSCAP if we didn't know how to find the closed form  $Z^*$  optimal for its dual. We will also show in this subsection why we call our solution method *projected coordinate descent* instead of *projected gradient descent* as in its original name.

### 4.3.1 Exercise: Dual Projected Coordinate Descent Solution for the LSCAP

In this section we will assume that we are not able to find a closed form  $Z^*$  while solving the dual (4.4) of the LSCAP (4.1) to demonstrate the application of our coordinate descent method. Recall that the LSCAP (4.1) was:

$$\begin{aligned} \min \quad & \frac{1}{2} \|X - G\|_F^2 \\ \text{such that} \quad & X \succeq 0, \\ & \text{Tr } A_i P = b_i, \quad j = 1, \dots, p, \\ & \text{Tr } C_j P \leq d_j, \quad j = 1, \dots, m. \end{aligned}$$

#### The Dual Problem and Properties

The  $X$  minimizing the Lagrangian (4.2) of the LSCAP (4.1) was given by (4.3):

$$X_m(Z, \nu, \mu) = G - A(\nu) - C(\mu) + Z,$$

which by plugging in the Lagrangian (4.2) yielded the following dual problem (4.4) of the LSCAP:

$$\begin{aligned} \text{maximize} \quad & g(Z, \nu, \mu) \quad \text{such that } Z \succeq 0, \mu \geq 0, \\ \text{where} \quad & g(Z, \nu, \mu) = -\frac{1}{2} \|G - A(\nu) - C(\mu) + Z\|_F^2 + \frac{1}{2} \|G\|_F^2 - \nu^T b - \mu^T d, \\ \text{or, in terms of } X_m, \quad & g(Z, \nu, \mu) = -\frac{1}{2} \|X_m\|_F^2 + \frac{1}{2} \|G\|_F^2 - \nu^T b - \mu^T d, \end{aligned}$$

for which weak duality always held and for which strong duality also held if the LSCAP (4.1) was strictly feasible. Note that  $X_m$  is symmetric since all of the matrices involved in it are symmetric. Note also that  $Z^*$ ,  $\nu^*$ , and  $\mu^*$  which are

the maximizer Lagrange multipliers (satisfying their own constraints) of this dual problem gave the optimal solution  $X^*$  of the LSCAP (4.1) via  $X_m$ , i.e.,  $X^* = X_m(Z^*, \nu^*, \mu^*)$ .

### The Critical Assumption

One of the critical points in [26] is that it is possible to analytically maximize  $g$  over the variable  $Z$  within the constraint  $Z \succeq 0$  by choosing

$$Z^* = (G - A(\nu) - C(\mu))_-,$$

which simplifies the dual problem (4.4) (to (4.6)). Now suppose that we weren't able to find such a closed form for  $Z^*$ . Then we wouldn't be able to simplify the dual problem (4.4) and would have to calculate the gradients of the dual objective from this dual function  $g$ . Not only the gradients with respect to  $\mu$  and  $\nu$  but now also the gradient with respect to  $Z$  need to be calculated.

### Finding the Gradients of the Dual Objective with respect to the Lagrange multipliers

Using the numerator layout, let us first find the gradient of  $g$  with respect to  $X_m$ :

$$\frac{\partial g}{\partial X_m} = \frac{\partial \left( -\frac{1}{2} \|X_m\|_F^2 \right)}{\partial X_m} = -X_m^T = -X_m, \quad (4.15)$$

since  $X_m$  is symmetric. Since  $X_m$  is a linear function of  $Z$ ,  $A(\nu)$ , and  $C(\mu)$  and they appear in  $g$  via only  $X_m$ , the gradients of  $g$  with respect to  $Z$ ,  $A(\nu)$ , and  $C(\mu)$  can be found by a simple chain rule:

$$\begin{aligned} \frac{\partial g}{\partial Z} &= \frac{\partial g}{\partial X_m} \frac{\partial X_m}{\partial Z} = -X_m I = -X_m, \\ \frac{\partial g}{\partial A(\nu)} &= \frac{\partial g}{\partial X_m} \frac{\partial X_m}{\partial A(\nu)} = (-X_m)(-I) = X_m, \\ \frac{\partial g}{\partial C(\mu)} &= \frac{\partial g}{\partial X_m} \frac{\partial X_m}{\partial C(\mu)} = (-X_m)(-I) = X_m. \end{aligned}$$

Since  $\nu_i$ 's and  $\mu_j$ 's are also involved in the dot products  $\nu^T b$  and  $\mu^T d$  respectively in  $g$ , these dot products should be accounted for while calculating the gradients of  $g$  with respect to  $\nu_i$  and  $\mu_j$ . Also using the chain rule (A.22) for scalar by scalar derivative involving matrices

$$\frac{\partial g(U)}{\partial x} = \mathbf{Tr} \left[ \frac{\partial g(U)}{\partial U} \frac{\partial U}{\partial x} \right],$$

---

**Algorithm 3** Dual Projected Coordinate Descent Solution for the LSCAP

---

**Init:** Set  $\mu = \nu = 0$ ,  $Z = 0$ , pick step-size parameter  $\alpha$

Compute  $C(\mu) = \sum_{j=1}^{N(N-1)} \mu_j C_j$  and  $A(\nu) = \sum_{i=1}^N \nu_i A_i$

**repeat**

Set  $X_{k,1} = G - C(\mu) - A(\nu) + Z$

Compute gradients  $\partial g / \partial \mu_j = \text{trace}(C_j X_{k,1})$ , let  $\mu_j = (\mu_j + \alpha \text{trace}(C_j X_{k,1}))_+$

Compute  $C(\mu) = \sum_{j=1}^{N(N-1)} \mu_j C_j$

Set  $X_{k,2} = G - C(\mu) - A(\nu) + Z$

Compute gradients  $\partial g / \partial \nu_i = \text{trace}(A_i X_{k,2})$ , let  $\nu_i = \nu_i + \alpha \text{trace}(A_i X_{k,2})$

Compute  $A(\nu) = \sum_{i=1}^N \nu_i A_i$

Set  $X_{k,3} = G - C(\mu) - A(\nu) + Z$

Compute gradient  $\partial g / \partial Z = -X_{k,3}$ , let  $Z = (Z + \alpha(-X_{k,3}))_+$

**until** convergence

$X^* = (G - C(\mu) - A(\nu) + Z)_+$ .

---

these gradients are

$$\begin{aligned} \frac{\partial g}{\partial \nu_i} &= \mathbf{Tr} \left[ \frac{\partial g}{\partial A(\nu)} \frac{\partial A(\nu)}{\partial \nu_i} \right] + \frac{\partial(-\nu^T b)}{\partial \nu_i} = \mathbf{Tr} [X_m A_i^T] - b_i = \mathbf{Tr} [X_m A_i] - b_i, \\ \frac{\partial g}{\partial \mu_j} &= \mathbf{Tr} \left[ \frac{\partial g}{\partial C(\mu)} \frac{\partial C(\mu)}{\partial \mu_j} \right] + \frac{\partial(-\mu^T d)}{\partial \mu_j} = \mathbf{Tr} [X_m C_j^T] - d_j = \mathbf{Tr} [X_m C_j] - d_j, \end{aligned}$$

### The Algorithm

We can finally present our dual projected coordinate descent algorithm for the LSCAP in Algorithm 3. This algorithm has three differences from its original gradient projection counterpart, first two of which are major and the last of which is minor:

1. The intermediate  $X$ 's are no more projected onto the positive semidefinite cone.
2. Since now updating  $X$  has a very small cost, we update it after descent in each of  $\mu$ ,  $\nu$ , and  $Z$ . Therefore, we call our algorithm a *coordinate descent* instead of a *gradient descent*.
3. There is now a final required step after the iterations are finished: Since we no longer project the intermediate  $X$ 's onto the p.s.d. cone, we project it at the end, but totaling to only once in the algorithm (It should be noted, however, that this operation brings a very minor additional cost since we are projecting  $Z$  at each iteration.). We need to do this operation since however much the intermediate  $X_{K,3}$  calculated in the last iteration  $K$  may be close to the real p.s.d.  $X^*$ , there is always the possibility that  $X_{K,3}$  is in fact not p.s.d..



There are of course many similarities to the gradient projection method, and most important of them is the complexity of the projected coordinate descent algorithm. Although the intermediate  $X$  is no more projected onto the p.s.d. cone, now  $Z$  is projected onto the p.s.d. cone in its update stage once in every iteration. Therefore, there is still one operation of projection onto the p.s.d. cone in every iteration. Since the main complexity of both of the algorithms lies in the evaluation of the SVD needed for these projections, the two algorithms have similar complexity.

### 4.3.2 Solution for the Smooth and Monotone Problem

Now we turn back once again to our smooth and monotone problem (3.8):

$$\begin{aligned} \min \quad & \frac{1}{2} \|P - \hat{P}\|_F^2 + \lambda \|M_s \circ (D_s P)\|_F^2 \\ \text{such that} \quad & P \succeq 0, \quad M_{m,l} \circ (D_{m,l} P) + M_{m,u} \circ (D_{m,u} P) \geq 0 \end{aligned}$$

which we also expressed in (4.14) as

$$\begin{aligned} \min \quad & \frac{1}{2} \|P - \hat{P}\|_F^2 + \lambda \|M_s \circ (D_s P)\|_F^2 \\ \text{such that} \quad & P \succeq 0, \\ & \mathbf{Tr} C_j P \leq 0, \quad j = 1, \dots, N(N-1). \end{aligned}$$

which we noted that is not fully in the LSCAP form (4.1) due to the extra second (smoothing) term in the objective, explaining how this term changes the solution of the problem with its dual.

#### Modification in the Objective

The Lagrangian of the problem (4.14) can be obtained easily with a modification on the Lagrangian  $L$  (4.2) of the LSCAP (4.1) through the addition of the extra smoothing term and the removal of the elements related to  $\nu$ . Then we find the Lagrangian of the problem (4.14) as

$$\tilde{L}_{\text{sm}}(P, Z, \mu) = \frac{1}{2} \|P - \hat{P}\|_F^2 + \lambda \|M_s \circ (D_s P)\|_F^2 + \mathbf{Tr} P(-Z + C(\mu)) - \mu^T d, \quad (4.16)$$

which has one problem: The  $P$  that minimizes the Lagrangian  $\tilde{L}_{\text{sm}}$  does not necessarily have symmetry property intrinsically, a property from which we benefited in the derivation of the solution of the LSCAP (4.1). For this reason we make a slight change in the expression of (4.14) and express the *smooth and monotone problem* from now on as

$$\begin{aligned} \text{minimize} \quad & f_{\text{sm}}(P) \\ \text{where} \quad & f_{\text{sm}}(P) = \frac{1}{2} \|P - \hat{P}\|_F^2 + \frac{\lambda}{2} \|M_s \circ (D_s P)\|_F^2 + \frac{\lambda}{2} \|M_s \circ (D_s P^T)\|_F^2, \\ \text{such that} \quad & P \succeq 0, \\ & \mathbf{Tr} C_j P \leq 0, \quad j = 1, \dots, N(N-1), \end{aligned} \quad (4.17)$$

This formulation is equivalent to (4.14) since the sum of the second and third terms in the objective of (4.17), i.e.,  $\frac{\lambda}{2}\|M_s \circ (D_s P)\|_F^2 + \frac{\lambda}{2}\|M_s \circ (D_s P^T)\|_F^2$  is equal to the second term in the objective of (4.14), i.e.,  $\lambda\|M_s \circ (D_s P)\|_F^2$  due to the symmetry property of  $P$  from the positive semi-definite constraint  $P \succeq 0$ .

### Finding the Minimizer $P$ of the Lagrangian

The Lagrangian of this problem (4.17) can be again obtained easily:

$$L_{\text{sm}}(P, Z, \mu) = \frac{1}{2}\|P - \hat{P}\|_F^2 + \frac{\lambda}{2}\|M_s \circ (D_s P)\|_F^2 + \frac{\lambda}{2}\|M_s \circ (D_s P^T)\|_F^2 + \text{Tr}P(-Z + C(\mu)) - \mu^T d, \quad (4.18)$$

for which we make the following claim and prove that claim.

**Claim.**  $L_{\text{sm}}$  has a unique and symmetric minimizer  $P_{\text{sm}}$ .

**Proof.** (i) *Uniqueness:* Since  $f_{\text{sm}}$  (the objective of (4.17)) and the constraint set are convex, at least one minimizer exists. Call it  $P_{\text{sm}}$ .

Now suppose that there exists  $Y$  such that  $L_{\text{sm}}(Y) = L_{\text{sm}}(P_{\text{sm}})$  but  $Y \neq P_{\text{sm}}$ . Then, however,  $W = (P_{\text{sm}} + Y)/2$  both satisfies all of the constraints and  $f_{\text{sm}}(W) < f_{\text{sm}}(P_{\text{sm}}) = f_{\text{sm}}(Y)$ , since  $f_{\text{sm}}$  is a summation of strictly convex and linear terms.

Hence contradiction.

(ii) *Symmetry:* Suppose  $P_{\text{sm}}$  is not symmetric, i.e.,  $P_{\text{sm}}^T \neq P_{\text{sm}}$ . However,  $P_{\text{sm}}^T$  satisfies all of the constraints and  $L(P_{\text{sm}}^T) = L(P_{\text{sm}})$ , and due to the strict convexity of  $L_{\text{sm}}$ ,  $L_{\text{sm}}((P_{\text{sm}} + P_{\text{sm}}^T)/2) < L_{\text{sm}}(P_{\text{sm}})$ .

Hence contradiction. ■

We can now start solving the smooth and monotone problem (4.17). We first need to find  $P_{\text{sm}}$  in a closed expression. To that end, we first transform  $P$  into a vector  $\text{vec}(P)$  by stacking its columns, i.e., by the *stacking* operation:

$$\text{vec}(X) = \sum_{i=1}^N I_{N^2 \times N}^i X I_{N \times 1}^i, \quad (4.19)$$

where

$$[I_{N^2 \times N}^i]_{(k,l)} = \begin{cases} 1, & \text{if } k=l+(i-1)N \\ 0, & \text{otherwise} \end{cases} \quad \text{and} \quad [I_{N \times 1}^i]_k = \begin{cases} 1, & \text{if } k=i \\ 0, & \text{otherwise} \end{cases},$$

i.e.,  $I_{N^2 \times N}^i$  is an  $N^2 \times N$  matrix of zeros except that its  $i^{\text{th}}$  block is an identity matrix, and  $I_{N \times 1}^i$  is a  $N \times 1$  vector of zeros except that its  $i^{\text{th}}$  entry is 1.

We can now define  $(N-2)(N-1) \times N^2$  matrices  $D_{s1}$  and  $D_{s2}$  such that

$$\|M_s \circ (D_s P)\|_F^2 = \|D_{s1} \text{vec}(P)\|_F^2 \quad \text{and} \quad \|M_s \circ (D_s P^T)\|_F^2 = \|D_{s2} \text{vec}(P)\|_F^2. \quad (4.20)$$

We already know that  $D_s$  is a  $(N-2) \times N$  matrix and that  $M_s$  multiplies  $(N-2)$  entries of  $D_s P$  with zero. This is the reason the size of  $(N-2)(N-1) \times N^2$  is sufficient for  $D_{s1}$  and  $D_{s2}$ .

To calculate the gradient of the Lagrangian  $L_{\text{sm}}$  with respect to  $X$ , we will need to find the gradients of smoothing terms (4.20) with respect to  $P$ . For that, we first express these smoothing terms in a simpler, trace function form:

$$\begin{aligned} \|M_s \circ (D_s P)\|_F^2 &= \|D_{s1} \text{vec}(P)\|_F^2 = \left\| \sum_{i=1}^N D_{s1} I_{N^2 \times N}^i P I_{N \times 1}^i \right\|_F^2 \\ &= \text{Tr} \left[ \left( \sum_{i=1}^N I_{N \times 1}^i P^T I_{N^2 \times N}^i D_{s1}^T \right) \left( \sum_{j=1}^N D_{s1} I_{N^2 \times N}^j P I_{N \times 1}^j \right) \right] \\ &= \sum_{i=1}^N \sum_{j=1}^N \text{Tr} [I_{N \times 1}^j I_{N \times 1}^i P^T I_{N^2 \times N}^i D_{s1}^T D_{s1} I_{N^2 \times N}^j P] = \sum_{i=1}^N \sum_{j=1}^N \text{Tr} [N_{j,i} P^T R_{i,j} P], \end{aligned}$$

where we used the fact that the trace function allows cyclic permutation. Here,  $N_{i,j} = I_{N \times 1}^i I_{N \times 1}^j T$ , i.e., an  $N \times N$  matrix of zeros with only its  $(i,j)^{\text{th}}$  entry 1, and  $R_{i,j} = I_{N^2 \times N}^i D_{s1}^T D_{s1} I_{N^2 \times N}^j$ , i.e.,  $(i,j)^{\text{th}}$   $N \times N$  block of  $N^2 \times N^2$  matrix  $D_{s1}^T D_{s1}$ . An important property of these matrices which we will use frequently is that  $N_{i,j}^T = N_{j,i}$  and that  $R_{i,j}^T = R_{j,i}$  (the latter is due to the symmetry of  $D_{s1}^T D_{s1}$ ).

Now, using the identity (A.23) from matrix calculus

$$\frac{\partial \text{Tr}[BX^T AX]}{\partial X} = B^T X^T A^T + BX^T A$$

we find

$$\begin{aligned} \frac{\partial \|M_s \circ (D_s P)\|_F^2}{\partial P} &= \frac{\partial \sum_{i=1}^N \sum_{j=1}^N \text{Tr} [N_{j,i} P^T R_{i,j} P]}{\partial P} \\ &= \sum_{i=1}^N \sum_{j=1}^N \left( \underbrace{N_{j,i}^T}_{N_{i,j}^T} P^T \underbrace{R_{i,j}^T}_{R_{j,i}^T} + N_{j,i} P^T R_{i,j} \right) = 2 \sum_{i=1}^N \sum_{j=1}^N N_{i,j} P^T R_{j,i} \end{aligned}$$

Similarly, if we let  $K_{i,j} = I_{N^2 \times N}^i D_{s2}^T D_{s2} I_{N^2 \times N}^j$ , i.e.,  $(i,j)^{\text{th}}$   $N \times N$  block of  $N^2 \times N^2$  matrix  $D_{s2}^T D_{s2}$ , then

$$\frac{\partial \|M_s \circ (D_s P^T)\|_F^2}{\partial P} = \frac{\partial \|D_{s2} \text{vec}(P)\|_F^2}{\partial P} = 2 \sum_{i=1}^N \sum_{j=1}^N N_{i,j} P^T K_{j,i},$$

where again  $K_{i,j}$  has the property  $K_{i,j}^T = K_{j,i}$  due to the symmetry of  $D_{s2}^T D_{s2}$ .

Now we can set the gradient of the Lagrangian  $L_{\text{sm}}$  to zero to find  $P_{\text{sm}}$ :

$$\begin{aligned} 0 &= \partial L_{\text{sm}} / \partial P \\ &= P_{\text{sm}}^T - \hat{P}^T + \frac{\lambda}{2} \left( 2 \sum_{i=1}^N \sum_{j=1}^N N_{i,j} P_{\text{sm}}^T R_{j,i} \right) + \frac{\lambda}{2} \left( 2 \sum_{i=1}^N \sum_{j=1}^N N_{i,j} P_{\text{sm}}^T K_{j,i} \right) - Z + C(\mu). \end{aligned} \tag{4.21}$$

Stacking columns, denoting

$$\tilde{G}(Z, \mu) = \hat{P} + Z - C(\mu) \quad (4.22)$$

using the symmetry of  $\hat{P}$ , and also using the formerly proven symmetry of  $P_{\text{sm}}$ , we can express the equation (4.21) as

$$\text{vec}(P_{\text{sm}}) + \text{vec} \left( \lambda \sum_{i=1}^N \sum_{j=1}^N N_{i,j} P_{\text{sm}}(R_{j,i} + K_{j,i}) \right) = \text{vec}(\tilde{G}). \quad (4.23)$$

Using the property that  $\text{vec}(ABC) = (C^T \otimes A)\text{vec}(B)$  where  $\otimes$  is the Kronecker product, we can simplify Equation (4.23) further as

$$\underbrace{\left( I + \lambda \sum_{i=1}^N \sum_{j=1}^N [(R_{j,i}^T + K_{j,i}^T) \otimes N_{i,j}] \right)}_{\tilde{B}(\lambda)} \text{vec}(P_{\text{sm}}) = \text{vec}(\tilde{G}). \quad (4.24)$$

The properties  $R_{i,j}^T = R_{j,i}$ ,  $K_{i,j}^T = K_{j,i}$ , and  $N_{i,j}^T = N_{j,i}$  makes the  $N^2 \times N^2$  matrix  $\tilde{B}(\lambda)$  symmetric. Moreover, since we have proved that  $P_{\text{sm}}$  is unique,  $\tilde{B}$  is invertible. Therefore,  $\text{vec}(P_{\text{sm}})$  is given by

$$\text{vec}(P_{\text{sm}}) = \tilde{B}^{-1} \text{vec}(\tilde{G}), \quad (4.25)$$

where  $\tilde{B}^{-1}$  is symmetric since  $\tilde{B}$  is.

We need to transform  $\text{vec}(P_{\text{sm}})$  back into matrix form  $P_{\text{sm}}$  to derive the dual function. Therefore, this time we use the *unstacking* operation

$$\text{mat}(\text{vec}(X)) = \sum_{i=1}^N I_{N \times N^2}^i \text{vec}(X) I_{1 \times N}^i, \text{ where } I_{N \times N^2}^i = I_{N^2 \times N}^{i,T} \text{ and } I_{1 \times N}^i = I_{N \times 1}^{i,T}, \quad (4.26)$$

on  $\text{vec}(P_{\text{sm}})$  which is defined by (4.25), i.e., unstack the columns of  $\text{vec}(P_{\text{sm}})$ :

$$\begin{aligned} P_{\text{sm}} &= \text{mat}(\text{vec}(P_{\text{sm}})) = \text{mat}(\tilde{B}^{-1} \text{vec}(\tilde{G})) = \sum_{i=1}^N I_{N \times N^2}^i \tilde{B}^{-1} \text{vec}(\tilde{G}) I_{1 \times N}^i \\ &= \sum_{i=1}^N I_{N \times N^2}^i \tilde{B}^{-1} \left( \sum_{j=1}^N I_{N^2 \times N}^j \tilde{G} I_{N \times 1}^j \right) I_{1 \times N}^i = \sum_{i=1}^N \sum_{j=1}^N I_{N \times N^2}^i \tilde{B}^{-1} I_{N^2 \times N}^j \tilde{G} \overbrace{I_{N \times 1}^j I_{1 \times N}^i}^{N_{j,i}} \end{aligned}$$

Defining  $M_{i,j}(\lambda) = I_{N \times N^2}^i (\tilde{B}^{-1}(\lambda)) I_{N^2 \times N}^j$ , i.e.,  $(i,j)^{\text{th}}$   $N \times N$  block of  $N^2 \times N^2$  matrix  $\tilde{B}^{-1}$ , this gives:

$$P_{\text{sm}} = \sum_{i=1}^N \sum_{j=1}^N M_{i,j} \tilde{G} N_{j,i}, \quad (4.27)$$

where also  $M_{i,j}$  has the property that  $M_{i,j}^T = M_{j,i}$ .

## The Dual Problem and Properties

Now we can turn back to the Lagrangian  $L_{\text{sm}}$  and plug in  $P_{\text{sm}}$  to find the dual function  $g_{\text{sm}}(Z, \mu) = L_{\text{sm}}(P_{\text{sm}}, Z, \mu)$ :

$$g_{\text{sm}}(Z, \mu) = \frac{1}{2} \|P_{\text{sm}} - \hat{P}\|_F^2 + \frac{\lambda}{2} \|M_{\text{s}} \circ (D_{\text{s}} P_{\text{sm}})\|_F^2 + \frac{\lambda}{2} \|M_{\text{s}} \circ (D_{\text{s}} P_{\text{sm}}^T)\|_F^2 + \text{Tr} P_{\text{sm}}(-Z + C(\mu)) - \mu^T d, \quad (4.28)$$

Substituting the first term with its quadratic expansion, i.e., with

$$\frac{1}{2} \|P_{\text{sm}} - \hat{P}\|_F^2 = \frac{1}{2} \|P_{\text{sm}}\|_F^2 - \text{Tr}[P_{\text{sm}}^T \hat{P}] + \frac{1}{2} \|\hat{P}\|_F^2 \quad (4.29)$$

and exploiting the symmetry of  $P_{\text{sm}}$ , the dual function  $g_{\text{sm}}$  in (4.28) becomes

$$g_{\text{sm}}(Z, \mu) = \frac{1}{2} \|P_{\text{sm}}\|_F^2 - \text{Tr}[P_{\text{sm}} \hat{P}] + \frac{1}{2} \|\hat{P}\|_F^2 + \lambda \|M_{\text{s}} \circ (D_{\text{s}} P_{\text{sm}})\|_F^2 + \text{Tr} P_{\text{sm}}(-Z + C(\mu)) - \mu^T d, \quad (4.30)$$

hence using our notation  $\tilde{G}(Z, \mu) = \hat{P} + Z - C(\mu)$ , the dual problem becomes

$$\begin{aligned} \max \quad & g_{\text{sm}}(Z, \mu) \quad \text{such that} \quad Z \succeq 0, \quad \mu \geq 0, \\ \text{where} \quad & g_{\text{sm}}(Z, \mu) = \frac{1}{2} \|P_{\text{sm}}\|_F^2 + \lambda \|M_{\text{s}} \circ (D_{\text{s}} P_{\text{sm}})\|_F^2 - \text{Tr}[P_{\text{sm}} \tilde{G}] + \frac{1}{2} \|\hat{P}\|_F^2 - \mu^T d. \end{aligned} \quad (4.31)$$

Again, weak duality always holds for this problem. Strong duality also holds since the smooth and monotone problem (4.17) is strictly feasible, i.e., there exists a  $P \succ 0$  that satisfies the inequalities in (4.17): Take once again  $P = I$ , which satisfied the inequalities in (4.10), which had the same inequalities with (4.17). Therefore there exist  $Z^*$  and  $\mu^*$  that are optimal for this dual problem with dual objective and that give the optimal solution  $P^*$  of the smooth and monotone problem (4.17) via  $P_{\text{sm}}$ , i.e., via  $P^* = P_{\text{sm}}(Z^*, \mu^*)$ .

## Finding the Gradients of the Dual Objective with respect to the Lagrange multipliers

Here, we refer the reader to Appendix B for the derivation of the gradients

$$\frac{\partial g_{\text{sm}}}{\partial Z} = \text{mat} \left\{ M_{\tilde{G}} \text{vec}(\tilde{G}) \right\}, \quad (4.32)$$

$$\frac{\partial g_{\text{sm}}}{\partial \mu_j} = -\text{Tr} \left[ \text{mat} \left\{ M_{\tilde{G}} \text{vec}(\tilde{G}) \right\} C_j \right], \quad (4.33)$$

where  $M_{\tilde{G}}$  is an  $N^2 \times N^2$  matrix and a function of  $\lambda$  given in (B.5) as

$$M_{\tilde{G}}(\lambda) = \sum_{i=1}^N \sum_{j=1}^N \left( -2 [N_{j,i}^T \otimes M_{i,j}(\lambda)] + \sum_{l=1}^N \left[ (M_{i,j}(\lambda))^T \underbrace{(E_{i,l}(\lambda))^T}_{M_{l,i}(\lambda) + 2\lambda \sum_{k=1}^N M_{l,k}(\lambda) R_{k,i}} \otimes N_{j,l} \right] \right) \quad (4.34)$$

---

**Algorithm 4** Dual Projected Coordinate Descent Solution for the Smooth and Monotone Problem

---

**Init:** Set  $\mu=0$ ,  $Z=0$ , pick step-size parameter  $\alpha$ , compute  $C(\mu) = \sum_{j=1}^{N(N-1)} \mu_j C_j$   
**repeat**  
    Set  $\tilde{G}_{k,1} = \hat{P} - C(\mu) + Z$   
    Compute gradients  $\frac{\partial g_{\text{sm}}}{\partial \mu_j} = -\text{Tr} \left( \text{mat}\{M_{\tilde{G}} \text{vec}(\tilde{G})\} C_j \right)$ , let  $\mu_j = \left( \mu_j + \alpha \frac{\partial g_{\text{sm}}}{\partial \mu_j} \right)_+$   
    Compute  $C(\mu) = \sum_{j=1}^{N(N-1)} \mu_j C_j$   
    Set  $\tilde{G}_{k,2} = \hat{P} - C(\mu) + Z$   
    Compute gradient  $\frac{\partial g_{\text{sm}}}{\partial Z} = \text{mat}\{M_{\tilde{G}} \text{vec}(\tilde{G})\} C_j$ , let  $Z = \left( Z + \alpha \frac{\partial g_{\text{sm}}}{\partial Z} \right)_+$   
**until** convergence  
 $P_{\text{sm}} = \sum_{i=1}^N \sum_{j=1}^N M_{i,j} (\hat{P} - C(\mu) + Z) N_{j,i}$ ,  $P^* = (P_{\text{sm}})_+$ .

---

### The Algorithm

We can finally present our dual projected coordinate descent algorithm for the smooth and monotone problem in Algorithm 4. This algorithm has six major differences from the dual gradient projection algorithm for the LSCAP, or, for the monotone problem (i.e. Algorithm 1), the last three of which are similar to those explained in the last subsection during the comparison of Algorithm 3 with Algorithm 1, but first three of which are very different:

1. First of all, the two algorithms solve two different problems (i.e. problems with different objectives to be minimized). Specifically, as we will show in the Remarks thread at the end of this subsection, when we set  $\lambda=0$ , i.e., force the smoothing term in the objective to have no effect in Algorithm 4, then Algorithm 4 reduces to the version of Algorithm 3 with no equality constraints, i.e., no  $A_i$ . Since Algorithm 3 solves the same problem that Algorithm 1 does, in effect then Algorithm 4 reduces to Algorithm 1 for  $\lambda=0$ .
2. Another important point of Algorithm 4 is that the matrix that is needed to be updated is not the minimizer  $P_{\text{sm}}$  (4.27) of the Lagrangian  $L_{\text{sm}}$  (4.18) but it is  $\tilde{G}$ , which is much easier to compute than it is to compute  $P_{\text{sm}}$ , as opposed to all other algorithms we have covered, all of which require the minimizer of the Lagrangian to be updated in each iteration. This difference also brings the following difference into play.
3. There is now a final required operation consisting of two steps after the iterations have finished, but the first of these steps is very different than the one required at the end of Algorithm 3. This step is to finally calculate from the converged  $\tilde{G}$  the minimizer  $P_{\text{sm}}$  (4.27) of the Lagrangian  $L_{\text{sm}}$  (4.18), where the converged  $\tilde{G}$  here yields the converged  $P_{\text{sm}}$ , which is very close to the optimal solution of

the smooth and monotone problem, its closeness being dependent on the error tolerance for convergence. The interesting point here is that this  $P_{\text{sm}}$  is evaluated only once and at the end of the algorithm. The second step of the final operation is a distinct difference and therefore is mentioned as a 6<sup>th</sup> difference below.

4. The matrix that is needed to be updated in each iteration, i.e.  $\tilde{G}$  is not required to be projected onto the positive semidefinite cone as part of this update operation as opposed to the update of  $P_k$  in Algorithm 1, where the projection of  $P_k$  onto the p.s.d. cone is a required step of the update operation in each iteration.
5. Since updating  $\tilde{G}$  has a very small cost, we update it sequentially after descent in each of  $\mu$  and  $Z$  as opposed to  $P_k$  in Algorithm 1, where  $P_k$  is only updated after simultaneous descents in  $\mu$  and  $Z$  due to the high cost of projection of the projection on the p.s.d. cone. Therefore, we call our algorithm *coordinate descent* instead of *gradient descent*.
6. Now the second step of the final operation after the iterations have finished is also required: We project the  $P_{\text{sm}}$  constructed in the first step of the final operation onto the p.s.d. cone. We need to do this operation since however much the intermediate  $P_{\text{sm}}$  calculated in the first step of the final operation may be close to the real p.s.d.  $P^*$ , there is always the possibility that  $P_{\text{sm}}$  is in fact not p.s.d.. It should be noted, however, that this projection would not be necessary if the algorithm converged to the matrix  $P^*$  it converges exactly.

There are of course many similarities to the gradient projection method, and most important of them is the complexity of the projected coordinate descent algorithm. Although the intermediate  $\tilde{G}$  is not projected onto the p.s.d. cone, this time  $Z$  is projected onto the p.s.d. cone in its update stage once in every iteration. Therefore, there is still one operation of projection onto the p.s.d. in every iteration. In spite of the *vec* and *mat* operations required twice in each iteration of Algorithm 4, the main complexity of both algorithms still lies in the evaluation of the SVD needed for these projections, and, therefore, the two algorithms have similar complexity.

### Modification with Additional Constraints

We stated in Chapter 3 that it is straightforward to add additional constraints, e.g.,  $[P]_{(i,i)} = 1$  to our smooth and monotone formulation when dealing with correlation coefficient matrices, or positivity of correlations. It is again straightforward to add such constraints to the smooth and monotone problem (4.17) and modify the solution algorithm we just derived accordingly. To set an example, we now assume that we want to add the constraint  $[P]_{(i,i)} = 1$ .

In the modified smooth and monotone problem, the constraint of  $[P]_{(i,i)} = 1$

again takes form as  $\mathbf{Tr} A_i P = 1$  for  $i = 1, \dots, N$ , where  $A_i$ 's are  $N \times N$  matrices of zeros except that  $[A_i]_{(i,i)} = 1$ . Hence the *modified smooth and monotone problem* becomes

$$\text{minimize } f_{\text{sm}}(P) \quad (4.35)$$

$$\text{where } f_{\text{sm}}(P) = \frac{1}{2} \|P - \hat{P}\|_F^2 + \frac{\lambda}{2} \|M_s \circ (D_s P)\|_F^2 + \frac{\lambda}{2} \|M_s \circ (D_s P^T)\|_F^2,$$

$$\text{such that } P \succeq 0,$$

$$\mathbf{Tr} A_i P = 1, \quad j = 1, \dots, N,$$

$$\mathbf{Tr} C_j P \leq 0, \quad j = 1, \dots, N(N-1),$$

to which we will also simply refer as the *smooth and monotone problem with additional constraints*.

The solution of this problem with its dual follows exactly the same path described until this thread in this subsection, this time including  $A_i$ 's and  $\nu_i$ 's with the same format  $C_j$ 's and  $\mu_j$ 's are in respectively. We are still able to use this solution method since the modified smooth and monotone problem (4.35) is also strictly feasible: Again let  $P = I$ , which satisfies all of the constraints in (4.35).

To be more specific, the slight changes the addition of  $A_i$ 's and  $\nu_i$ 's cause with the respect to the order in the derivation of the solution can be listed as:

1. The terms  $\mathbf{Tr} P A(\nu)$  and  $-\nu^T b$  are added to the Lagrangian  $L_{\text{sm}}$  (4.18).
2. The term  $-A(\nu)$  is added to the  $\tilde{G}(Z, \mu)$  (4.22) changing it to  $\tilde{G}(Z, \mu, \nu) = \hat{P} - C(\mu) - A(\nu) + Z$ , although  $\tilde{G}(Z, \mu, \nu)$  is used exactly the same in the rest of the derivation.
3. Due to the 1<sup>st</sup> change above, the terms  $\mathbf{Tr} P_{\text{sm}} A(\nu)$  and  $-\nu^T b$  are added to the dual function  $g_{\text{sm}}$  in the dual problem (4.31).
4. Due to the 2<sup>nd</sup> change above, the primal solution is now

$$\begin{aligned} P^* &= P_{\text{sm}}(Z^*, \mu^*, \nu^*) = \sum_{i=1}^N \sum_{j=1}^N M_{i,j} \tilde{G}(Z^*, \mu^*, \nu^*) N_{j,i} \\ &= \sum_{i=1}^N \sum_{j=1}^N M_{i,j} (\hat{P} - C(\mu^*) - A(\nu^*) + Z) N_{j,i}, \end{aligned} \quad (4.36)$$

where  $\nu^*$  comes into play as the optimal additional Lagrangian multiplier for the dual  $g_{\text{sm}}$  of the modified smooth and monotone problem.

5. Since  $\nu_i$ 's are now additional Lagrange multipliers, we need to calculate the gradient  $\partial g_{\text{sm}} / \partial \nu_i$  as well. The derivation of this gradient follows exactly that of  $\partial g_{\text{sm}} / \partial \mu_j$  through (B.8) and (B.9) with  $d_j$ ,  $\mu_j$ ,  $C_j$ , and  $C(\mu)$  replaced with  $b_i$ ,  $\nu_i$ ,  $A_i$ , and  $A(\nu)$  respectively. This replacement yields

$$\frac{\partial g_{\text{sm}}}{\partial \nu_i} = -\mathbf{Tr} \left[ \text{mat} \left\{ M_{\tilde{G}} \text{vec}(\tilde{G}) \right\} A_i \right] - 1, \quad (4.37)$$

as  $b_i = 1$  for all  $i = 1, \dots, N$ .



---

**Algorithm 5** Dual Projected Coordinate Descent Solution for the Smooth and Monotone Problem with Additional Constraints

---

**Init:** Set  $\mu = \nu = 0$ ,  $Z = 0$ , pick step-size parameter  $\alpha$

Compute  $C(\mu) = \sum_{j=1}^{N(N-1)} \mu_j C_j$  and  $A(\nu) = \sum_{i=1}^N \nu_i A_i$

**repeat**

Set  $\tilde{G}_{k,1} = \hat{P} - C(\mu) - A(\nu) + Z$

Compute gradient  $\frac{\partial g_{\text{sm}}}{\partial \mu_j} = -\text{Tr} \left( \text{mat}\{M_{\tilde{G}} \text{vec}(\tilde{G})\} C_j \right)$ , let  $\mu_j = \left( \mu_j + \alpha \frac{\partial g_{\text{sm}}}{\partial \mu_j} \right)_+$

Compute  $C(\mu) = \sum_{j=1}^{N(N-1)} \mu_j C_j$

Set  $\tilde{G}_{k,2} = \hat{P} - C(\mu) - A(\nu) + Z$

Compute gradient  $\frac{\partial g_{\text{sm}}}{\partial Z} = \text{mat}\{M_{\tilde{G}} \text{vec}(\tilde{G})\} C_j$ , let  $Z = \left( Z + \alpha \frac{\partial g_{\text{sm}}}{\partial Z} \right)_+$

Set  $\tilde{G}_{k,3} = \hat{P} - C(\mu) - A(\nu) + Z$

Compute gradients  $\frac{\partial g_{\text{sm}}}{\partial \nu_i} = -\text{Tr} \left( \text{mat}\{M_{\tilde{G}} \text{vec}(\tilde{G})\} A_i \right)$ , let  $\nu_i = \nu_i + \alpha \frac{\partial g_{\text{sm}}}{\partial \nu_i}$

Compute  $A(\nu) = \sum_{i=1}^N \nu_i A_i$

**until** convergence

$P_{\text{sm}} = \sum_{i=1}^N \sum_{j=1}^N M_{i,j} (\hat{P} - C(\mu) - A(\nu) + Z) N_{j,i}$ ,  $P_{\text{sm}}^* = (P_{\text{sm}})_+$ .

---

To find the new  $P^*(Z^*, \mu^*, \nu^*)$  via the gradients of  $g_{\text{sm}}$  with respect to  $Z$ ,  $\mu_j$ , and newly added  $\nu_i$  as in the unmodified projected coordinate descent algorithm (i.e. Algorithm 4), we implement in Algorithm 5 a slightly modified version reflecting the changes listed above. Specifically, the 2<sup>nd</sup>, 4<sup>rd</sup>, and 5<sup>th</sup> are the apparent changes directly affecting the implementation of the algorithm, whereas the other changes are the indirect ones causing the emergence of the apparent ones.

As a last point, all of the points made on the comparison of Algorithm 4 with Algorithm 1 are also valid for the comparison of Algorithm 5 with Algorithm 2.

### Smoothness of the Dual Function

Whether the dual function  $g_{\text{sm}}$  is smooth will be of importance in the next subsection, in which we will adapt FISTA to the algorithms we derived; therefore, we briefly prove it for the modified projected coordinate descent solution (Algorithm 5). Since the unmodified algorithm (Algorithm 4) is a reduced version of this algorithm, then the proof for the former is trivial.

Now recall that a function  $g : \mathbb{R}^n \rightarrow \mathbb{R}$  is smooth if there exists a Lipschitz constant  $L(g)$  for the gradient  $\nabla g$ , i.e., for the mapping from  $x$  to  $(\partial g / \partial x)$  such that

$$\|\nabla g(x) - \nabla g(y)\| \leq L(g) \|x - y\|$$

for every  $x, y \in \mathbb{R}^n$ .

In our case  $g = g_{\text{sm}}(Z, \mu, \nu)$ ,  $x = (Z_x, \mu_x, \nu_x)$  (i.e., the concatenation of  $Z$ ,  $\mu$ , and  $\nu$ ), and  $n = N^2 + N(N-1) + N$  (each term in the summation reflecting the total number of elements in  $Z$ ,  $\mu$ , and  $\nu$  respectively). Therefore, we should prove the existence of a Lipschitz constant  $L(g_{\text{sm}})$  for the gradient  $\nabla g_{\text{sm}}$ , i.e., for the mapping from  $(Z, \mu, \nu)$  to  $(\partial g_{\text{sm}}/\partial Z, \partial g_{\text{sm}}/\partial \mu, \partial g_{\text{sm}}/\partial \nu)$  such that

$$\|\nabla g_{\text{sm}}(Z_x, \mu_x, \nu_x) - \nabla g_{\text{sm}}(Z_y, \mu_y, \nu_y)\| \leq L(g_{\text{sm}})\|(Z_x, \mu_x, \nu_x) - (Z_y, \mu_y, \nu_y)\| \quad (4.38)$$

for every  $x = (Z_x, \mu_x, \nu_x), y = (Z_y, \mu_y, \nu_y) \in \mathbb{R}^n$ .

**Proof.** We can make use of a simple trick to make this proof much easier. The only terms in  $g_{\text{sm}}$  in (4.31) that incorporate  $Z$ ,  $\mu$ , or  $\nu$  not via  $\tilde{G}$  is  $-\mu^T d - \nu^T b$ , which is a linear element of  $g_{\text{sm}}$  and which is a linear transformation of  $\mu$  and  $\nu$ . This term, therefore, isn't a candidate to prevent to existence of a Lipschitz constant since

$$\|\nabla(\mu_x^T d + \nu_x^T b) - \nabla(\mu_y^T d + \nu_y^T b)\| = 0$$

and we can ignore this term from this regard. Moreover,  $\tilde{G}(Z, \mu, \nu) = \hat{P} - C(\mu) - A(\nu) + Z$  is a linear function of  $Z$ ,  $\mu$ , and  $\nu$ . Due to these two reasons, to prove the existence of a Lipschitz constant  $L(g_{\text{sm}})$  it is sufficient to prove the existence of another Lipschitz constant  $L_{\tilde{G}}(g_{\text{sm}})$  again for the gradient  $\nabla g_{\text{sm}}$  but this time for the mapping from  $\tilde{G}(x)$  to  $(\partial g_{\text{sm}}/\partial \tilde{G}(x))$  such that

$$\left\| \frac{\partial g_{\text{sm}}(x)}{\partial \tilde{G}(x)} - \frac{\partial g_{\text{sm}}(y)}{\partial \tilde{G}(y)} \right\| \leq L_{\tilde{G}}(g_{\text{sm}}) \|\tilde{G}(x) - \tilde{G}(y)\|$$

for any  $\tilde{G}(x), \tilde{G}(y)$  for every  $x, y \in \mathbb{R}^n$ . From the expression (B.6) for  $\partial g_{\text{sm}}/\partial \tilde{G}$  we can simplify this inequality to

$$\left\| \text{mat} \left\{ M_{\tilde{G}} \text{vec} \left( \tilde{G}(x) \right) \right\} - \text{mat} \left\{ M_{\tilde{G}} \text{vec} \left( \tilde{G}(y) \right) \right\} \right\| \leq L_{\tilde{G}}(g_{\text{sm}}) \|\tilde{G}(x) - \tilde{G}(y)\|$$

which if we apply  $\text{vec}$  operation to the inside of the norms on both sides since  $\text{vec}$  operation has no effect on the norm operation, combined with the linearity of  $\text{vec}$  operation and the fact that  $\text{vec}(\text{mat}(v)) = v$ , becomes

$$\left\| M_{\tilde{G}} \text{vec} \left( \tilde{G}(x) - \tilde{G}(y) \right) \right\| \leq L_{\tilde{G}}(g_{\text{sm}}) \left\| \text{vec} \left( \tilde{G}(x) - \tilde{G}(y) \right) \right\|.$$

Such a constant  $L_{\tilde{G}}(g_{\text{sm}})$  exists, which is equal to the maximum eigenvalue of  $M_{\tilde{G}}$ ; therefore,  $g_{\text{sm}}(Z, \mu, \nu)$  is a smooth function.  $\blacksquare$

The same proof can be used for the dual projected coordinate descent solution (Algorithm 4) for the smooth and monotone problem, i.e., the unmodified projected coordinate descent solution, where both  $g_{\text{sm}}$  and  $\tilde{G}$  are functions of  $(Z, \mu)$  instead of  $(Z, \mu, \nu)$ . This change in the expression of  $g_{\text{sm}}$  and  $\tilde{G}$  changes nothing in the flow of the above proof.

### Remark for the Case when $\lambda = 0$

We mentioned during the discussion of Algorithm 4 that when  $\lambda = 0$  Algorithm 4 reduces to the version of Algorithm 3 with no equality constraints, i.e., no  $A_i$ . Then this would mean that for  $\lambda = 0$  Algorithm 5, which is the version of Algorithm 4 with equality constraints, should reduce directly to Algorithm 3. We will now mathematically prove the latter, i.e., the reduction of Algorithm 5 to Algorithm 3 for  $\lambda = 0$ , which will automatically prove the former, i.e., the reduction of Algorithm 4 for  $\lambda = 0$  to the version of Algorithm 3 with no equality constraints.

**Proof.** Let us define  $I_K$  as a  $K \times K$  identity matrix for any  $K$ . When  $\lambda = 0$ ,  $\tilde{B}(\lambda) = I_{N^2}$  by its definition in (4.24). This makes  $\tilde{B}^{-1} = I_{N^2}$  as well, which further makes

$$M_{i,j}(\lambda)|_{\lambda=0} = I_{N \times N^2}^i(\tilde{B}^{-1}(0))I_{N^2 \times N}^j = I_{N \times N^2}^i I_{N^2} I_{N^2 \times N}^j = \begin{cases} I_N, & \text{if } j=i \\ 0, & \text{otherwise} \end{cases}$$

since the second expression from the left hand side above meant that  $M_{i,j}$  is  $(i,j)^{\text{th}}$   $N \times N$  block of  $N^2 \times N^2$  matrix  $\tilde{B}^{-1}$ . This further makes

$$E_{i,l}(\lambda)|_{\lambda=0} = M_{l,i}(0) + 2\lambda \sum_{k=1}^N M_{l,k}(0)R_{k,i} = M_{l,i}(0) = \begin{cases} I_N, & \text{if } l=i \\ 0, & \text{otherwise} \end{cases}$$

Now we can simplify  $\partial g_{\text{sm}}/\partial \tilde{G}$  given in (B.4):

$$\begin{aligned} \left. \frac{\partial g_{\text{sm}}}{\partial \tilde{G}} \right|_{\lambda=0} &= \sum_{i=1}^N \sum_{j=1}^N \left( -2M_{i,j}(0)\tilde{G}N_{j,i} + \sum_{l=1}^N N_{j,l}\tilde{G}E_{i,l}(0)M_{i,j}(0) \right) \\ &= \sum_{i=1}^N \left( -2\tilde{G}N_{i,i} + N_{i,i}\tilde{G} \right) = -2\tilde{G} \left( \sum_{i=1}^N N_{i,i} \right) + \left( \sum_{i=1}^N N_{i,i} \right) \tilde{G} \\ &= -2\tilde{G}I_N + I_N\tilde{G} = -\tilde{G} \end{aligned}$$

Now, the important points are:

1.  $\tilde{G}(Z, \nu, \mu) = \hat{P} - C(\mu) - A(\nu) + Z$  is exactly the same for  $G = \hat{P}$  with  $X_m(Z, \nu, \mu) = G - C(\mu) - A(\nu) + Z$  which was defined in (4.3) and which yielded  $\partial g/\partial X_m = -X_m$  (in (4.15)) during the derivation of Algorithm 3 in Subsection 4.3.1. Therefore,  $\partial g_{\text{sm}}/\partial \tilde{G}|_{\lambda=0}$  and  $\partial g/\partial X_m$  are exactly the same. (Here,  $g$  is the dual objective of the problem (4.4) Algorithm 3 solves.)
2. During the derivation of Algorithm 3,  $\partial g/\partial X_m$  is used to calculate the gradients  $\partial g/\partial Z$ ,  $\partial g/\partial \mu_j$ , and  $\partial g/\partial \nu_i$  exactly the same way  $\partial g_{\text{sm}}/\partial \tilde{G}$  is used to calculate the gradients  $\partial g_{\text{sm}}/\partial Z$ ,  $\partial g_{\text{sm}}/\partial \mu_j$ , and  $\partial g_{\text{sm}}/\partial \nu_i$  during the derivation of Algorithm 5. Since  $\partial g/\partial X_m$  and  $\partial g_{\text{sm}}/\partial \tilde{G}$  are the sole

determiners of these gradients respectively and  $\partial g/\partial X_m = \partial g_{\text{sm}}/\partial \tilde{G}|_{\lambda=0}$ , the corresponding gradients used in both algorithms are exactly equal, i.e.:

$$\frac{\partial g}{\partial Z} = \frac{\partial g_{\text{sm}}}{\partial Z} \Big|_{\lambda=0}, \quad \frac{\partial g}{\partial \mu_j} = \frac{\partial g_{\text{sm}}}{\partial \mu_j} \Big|_{\lambda=0}, \quad \frac{\partial g}{\partial \nu_i} = \frac{\partial g_{\text{sm}}}{\partial \nu_i} \Big|_{\lambda=0} \quad (4.39)$$

This equivalence makes the iterations of both algorithms exactly the same.

3. The only thing that is left to show is that the final operations done after the iterations have finished in both algorithms are equivalent. For  $\lambda = 0$ , the final operation of Algorithm 5 becomes:

$$\begin{aligned} & \left( \sum_{i=1}^N \sum_{j=1}^N M_{i,j}(0)(\hat{P} - C(\mu) - A(\nu) + Z)N_{j,i} \right)_+ \\ &= \left( \sum_{i=1}^N (\hat{P} - C(\mu) - A(\nu) + Z)N_{i,i} \right)_+ \\ &= \left( (\hat{P} - C(\mu) - A(\nu) + Z) \sum_{i=1}^N N_{i,i} \right)_+ = (G - C(\mu) - A(\nu) + Z)_+ \Big|_{G=\hat{P}}, \end{aligned}$$

i.e., the final operation of Algorithm 3 given  $G = \hat{P}$ . ■

## 4.4 Optimal First Order Methods with FISTA

The algorithms we have developed (i.e. Algorithms 1 - 5) avoid computing the Hessian, but unfortunately they are plagued by slow convergence, with error decreasing as  $O(1/k)$ , where  $k$  is the iteration number. However, Nesterov has shown in [24] that it is possible to obtain  $O(1/k^2)$  convergence for a multi-step first order method by a careful combination of the current and previous gradients. An extension of Nesterov's method to projected gradients was developed in [25], called FISTA.

In this section we optimize the first order methods we have derived previously in this Chapter, i.e., the dual projected gradient solution for the monotone problem derived in Section 4.2 and the dual projected coordinate descent solution for the smooth and monotone problem derived in Subsection 4.3.2. We do this optimization by developing faster versions of first Algorithm 2 in Subsection 4.4.2 and then of Algorithm 5 in Subsection 4.4.3, adapting FISTA to our problems. However, we start with revisiting FISTA in the next subsection for a quick recall.

### 4.4.1 FISTA Revisited

The main aim of Beck and Teboulle in [25] is to present a faster version of the class of Iterative Shrinkage-Thresholding Algorithms (ISTA). The general formulation of the problem in which the authors are interested in is

$$\min\{F(x) \triangleq g(x) + f(x) : x \in \mathbb{R}^n\}, \quad (4.40)$$

where  $f : \mathbb{R}^n \rightarrow \mathbb{R}^n$  is a continuous convex function which is possibly *nonsmooth* and  $g : \mathbb{R}^n \rightarrow \mathbb{R}^n$  is a smooth convex function with gradient which is Lipschitz continuous. That is, there exists a constant  $L(g)$  for which

$$\|\nabla g(x) - \nabla g(y)\| \leq L(g)\|x - y\|$$

for every  $x, y \in \mathbb{R}^n$ .

When  $F$  itself is a smooth convex function, i.e., when  $f(x) := 0$  making  $F = g$ , the general step of ISTA reduces to form of gradient descent, i.e.,

$$x_{k+1} = x_k - t_k \nabla g(x_k)$$

reducing ISTA to a gradient method. Letting  $t_k = 1/L(g)$  it is proved that ISTA in general has a worst-case complexity of  $O(1/k)$ , which is improved to  $O(1/k^2)$  by the FISTA with constant step size. The special case of this algorithm for the case when  $F = g$  is given below:

**Input:**  $L = L(g)$  - A Lipschitz constant of  $\nabla g$ .

**Step 0.** Take  $y_1 = x_0 \in \mathbb{R}^n$ ,  $t_1 = 1$ .

**Step k.** ( $k \geq 0$ ) Compute

- (a)  $x_k = y_k - \frac{1}{L} \nabla g(y_k)$ ,
- (b)  $t_{k+1} = (1 + \sqrt{1 + 4t_k^2})/2$ ,
- (c)  $y_{k+1} = x_k + \frac{t_k - 1}{t_{k+1}}(x_k - x_{k-1})$ ,

where the convergence of this algorithm is guaranteed for this step size of  $1/L$ . However, we make an important remark at the end of the next subsection about the use of Lipschitz constant this way.

#### 4.4.2 Optimal First Order Method for the Monotone Problem

In this subsection we adapt FISTA to our dual projected gradient solution we derived for the monotone problem in Section 4.2. To be specific, we adapt it to the solution of the problem with additional constraints (i.e., to Algorithm 2) since the adaptation to that without the additional constraints (i.e., to Algorithm 1) is then straightforward due to the fact that Algorithm 2 reduces to Algorithm 1 when all  $A_i$ 's are changed to zero matrices.

The dual objective  $\psi(\mu, \nu)$  (given in the dual problem (4.6)) of the monotone problem with additional constraints (4.12) is convex and Lipschitz continuous with the Lipschitz constant  $L = 4(N+1)(N-1)+N$  as the latter shown in (4.13). Since in addition the dual projected solution is obtained by maximizing this objective, we can adapt FISTA to this dual problem. Smooth function  $g$  of FISTA corresponds to  $-\psi$  of our solution (i.e., of Algorithm 2), and  $x$  of FISTA corresponds to  $(\mu, \nu)$

---

**Algorithm 6** Dual FISTA for the Monotone Problem with Additional Constraints

---

**Init:** Set  $\mu_{x_0} = \mu_y = \nu_{x_0} = \nu_y = 0$ , let step-size parameter  $\alpha = 1/L$ ,  $t_1 = 1$ .

**repeat**

    Compute  $t_{k+1} = (1 + \sqrt{1+4t_k^2})/2$

    Compute  $C(\mu_y) = \sum_{j=1}^{N(N-1)} (\mu_y)_j C_j$  and  $A(\nu_y) = \sum_{i=1}^N (\nu_y)_i A_i$

    Set  $P_k(y) = (\hat{P} - C(\mu_y) - A(\nu_y))_+$

    Compute gradients:  $\left(\frac{\partial \psi}{\partial \mu_j}\right)(y) = \text{Tr}(C_j P_k(y))$  and  $\left(\frac{\partial \psi}{\partial \nu_i}\right)(y) = \text{Tr}(A_i P_k(y))$

    Let  $(\mu_{x_k})_j = ((\mu_y)_j + \alpha \text{Tr}(C_j P_k(y)))_+$  and  $(\nu_{x_k})_i = (\nu_y)_i + \alpha \text{Tr}(A_i P_k(y))$

    Let  $(\mu_y)_j = (\mu_{x_k})_j + \frac{t_k-1}{t_{k+1}}((\mu_{x_k})_j - (\mu_{x_{k-1}})_j)$  and

$(\nu_y)_i = (\nu_{x_k})_i + \frac{t_k-1}{t_{k+1}}((\nu_{x_k})_i - (\nu_{x_{k-1}})_i)$

**until** convergence

---

of Algorithm 2 since  $\psi$  is a function of these variables. Therefore, the gradient  $\nabla g(x)$  of FISTA corresponds to  $-\nabla \psi(\mu, \nu) = -(\partial \psi / \partial \mu, \partial \psi / \partial \nu)$  in our Algorithm 2, finally enabling us to adapt FISTA to this algorithm. We present the adapted algorithm, i.e., the dual FISTA for the monotone problem with additional constraints in Algorithm 6. Note that we project  $\mu_{x_k}$  onto the positive orthant in each iteration since that is the constraint set of the dual problem (4.6).

### Important Remark Regarding the Use of the Lipschitz Constant(s)

One remark that should be made about this algorithm is that, as we will see in Section 4.5, in effect the step-size parameter choice of  $\alpha = 1/L$  is too small for practical use and yields a slow convergence rate, preventing us from taking the full advantage of FISTA. This is due to the large value of  $L$  which is on the order of  $O(N^2)$  (this is also valid for the case without additional constraints since then again  $L = 4(N+1)(N-1) \propto O(N^2)$  (4.11)). Therefore, again as we will show in Section 4.5, choosing a value for  $\alpha$  by trial and error so that it is sufficiently small not to prevent convergence in any trial yields a much faster convergence rate for Algorithm 6 than choosing  $\alpha = 1/L$  does.

The last fact means that we won't actually make use of the Lipschitz constant when implementing Algorithm 6, and, regarding the algorithm we will develop in the next subsection, it is the reason we didn't derive a Lipschitz constant (although we proved its existence) for the gradient of the dual objective  $g_{\text{sm}}$  of the smooth and monotone problem while proving its smoothness in Subsection 4.3.2 (the latter of which, however, is necessary to know how to adapt FISTA). Since the gradient  $\nabla g_{\text{sm}}(Z, \mu, \nu)$  (expressed through (4.32), (4.33), and (4.37)) involves much heavier operations on the variables  $(Z, \mu, \nu)$  it is mapped from than  $\nabla \psi(\mu, \nu)$  involves, the Lipschitz constant  $L_{\text{sm}}$  for  $\nabla g_{\text{sm}}$  is expected to be much larger than the one for  $\nabla \psi$ .

This corresponds to a much smaller step size  $\alpha$  when  $\alpha = 1/L_{\text{sm}}$  for the dual FISTA for the smooth and monotone problem we will derive in the next subsection, making the convergence rate too small to have any practical significance. Therefore, in Section 4.5 we will find an  $\alpha$  value for the dual FISTA for the smooth and monotone problem just like we will do for the one for the monotone problem, i.e., for Algorithm 6. For this reason, in the next subsection we won't be using a Lipschitz constant in the dual FISTA for the smooth and monotone problem when presenting the algorithm.

#### 4.4.3 Optimal First Order Method for the Smooth and Monotone Problem

After the remark made at the end of the previous subsection, we now proceed to adapt FISTA this time to our dual projected coordinate descent solution we derived for the smooth and monotone problem in Subsection 4.3.2. To be specific, we adapt it to the solution of the problem with additional constraints (i.e. to Algorithm 5) since the adaptation to that without the additional constraints (i.e. to Algorithm 4) is then again straightforward due to the fact that Algorithm 5 reduces to Algorithm 4 when all  $A_i$ 's are changed to zero matrices.

The flow of the adaptation is quite similar to that of we did in the previous subsection. The dual objective  $g_{\text{sm}}(Z, \mu, \nu)$  (given in the dual problem (4.31)) of the smooth and monotone problem with additional constraints (4.35) is convex and smooth as the latter proven in Subsection 4.3.2. Since in addition the dual projected solution is again obtained by maximizing this objective, we can adapt FISTA to this dual problem as well. Smooth function  $g$  of FISTA corresponds to  $-g_{\text{sm}}$  of our solution (i.e. of Algorithm 5), and  $x$  of FISTA corresponds to  $(Z, \mu, \nu)$  of Algorithm 5 since  $g_{\text{sm}}$  is a function of these variables. Therefore, the gradient  $\nabla g(x)$  of FISTA corresponds to  $-\nabla g_{\text{sm}}(Z, \mu, \nu) = -(\partial g_{\text{sm}}/\partial Z, \partial g_{\text{sm}}/\partial \mu, \partial g_{\text{sm}}/\partial \nu)$  in our Algorithm 5, finally enabling us to adapt FISTA to this algorithm as well. We present the adapted algorithm, i.e., the dual FISTA for the smooth and monotone problem with additional constraints in Algorithm 7.

#### Comparison between Algorithms 7 and 6

Although there are many similarities between the derivations of the two algorithms, Algorithms 7 and 6, there are a total of nine major differences between these adapted algorithms. Six of these differences are exactly the same with the differences between Algorithms 5 and 2, from which Algorithms 7 and 6 are adapted respectively, and were during their comparison in Subsection 4.3.2 referred to those between Algorithms 4 and 1, whose differences were listed previously in the same subsection. Now

---

**Algorithm 7** Dual FISTA for the Smooth and Monotone Problem with Additional Constraints

---

**Init:** Set  $\mu_{x_0} = \mu_y = \nu_{x_0} = \nu_y = 0$ ,  $Z_{x_0} = Z_y = 0$ , pick step-size parameter  $\alpha$   
 Let  $t_1 = 1$ , compute  $C(\mu_y) = \sum_{j=1}^{N(N-1)} (\mu_y)_j C_j$  and  $A(\nu_y) = \sum_{i=1}^N (\nu_y)_i A_i$   
**repeat**  
   Compute  $t_{k+1} = (1 + \sqrt{1+4t_k^2})/2$   
   Set  $\tilde{G}_{k,1} = \hat{P} - C(\mu_y) - A(\nu_y) + Z_y$   
   Compute gradients  $\left(\frac{\partial g_{\text{sm}}}{\partial \mu_j}\right)(y) = -\text{Tr}\left(\text{mat}\{M_{\tilde{G}} \text{vec}(\tilde{G})\} C_j\right)$   
   Let  $(\mu_{x_k})_j = \left((\mu_y)_j + \alpha \left(\frac{\partial g_{\text{sm}}}{\partial \mu_j}\right)(y)\right)_+$ , let  $(\mu_y)_j = (\mu_{x_k})_j + \frac{t_k-1}{t_{k+1}}((\mu_{x_k})_j - (\mu_{x_{k-1}})_j)$   
   Compute  $C(\mu_y) = \sum_{j=1}^{N(N-1)} (\mu_y)_j C_j$   
   Set  $\tilde{G}_{k,2} = \hat{P} - C(\mu_y) - A(\nu_y) + Z_y$   
   Compute gradient  $\left(\frac{\partial g_{\text{sm}}}{\partial Z}\right)(y) = \text{mat}\{M_{\tilde{G}} \text{vec}(\tilde{G})\}$   
   Let  $Z_{x_k} = (Z_y + \alpha \left(\frac{\partial g_{\text{sm}}}{\partial Z}\right)(y))_+$ , let  $Z_y = Z_{x_k} + \frac{t_k-1}{t_{k+1}}(Z_{x_k} - Z_{x_{k-1}})$   
   Set  $\tilde{G}_{k,3} = \hat{P} - C(\mu_y) - A(\nu_y) + Z_y$   
   Compute gradients  $\left(\frac{\partial g_{\text{sm}}}{\partial \nu_i}\right)(y) = -\text{Tr}\left(\text{mat}\{M_{\tilde{G}} \text{vec}(\tilde{G})\} A_i\right)$   
   Let  $(\nu_{x_k})_i = (\nu_y)_i + \alpha \left(\frac{\partial g_{\text{sm}}}{\partial \nu_i}\right)(y)$ , let  $(\nu_y)_i = (\nu_{x_k})_i + \frac{t_k-1}{t_{k+1}}((\nu_{x_k})_i - (\nu_{x_{k-1}})_i)$   
   Compute  $A(\nu_y) = \sum_{i=1}^N (\nu_y)_i A_i$   
**until** convergence  
 $P_{\text{sm}} = \sum_{i=1}^N \sum_{j=1}^N M_{i,j} (\hat{P} - C(\mu_y) - A(\nu_y) + Z_y) N_{j,i}$ ,  $P_{\text{sm}}^* = (P_{\text{sm}})_+$ .

---

we list the other three (distinct) differences.

The first of the three distinct differences is that this time we project in each iteration not only  $\mu_{x_k}$  onto the positive orthant but also  $Z_{x_k}$  onto the positive semidefinite cone due to the constraint set of the dual problem (4.31). The second one is more profound and is due to the fact that while Algorithm 6 is derived from a gradient method in which  $P_k$  is only updated after simultaneous descents in  $\mu$  and  $Z$ , Algorithm 7 is derived from a coordinate descent algorithm in which  $\tilde{G}$  is updated sequentially after descent in each of  $Z$ ,  $\mu$  and  $\nu$ : In Algorithm 7 FISTA is applied in each of these sequential descent steps separately as opposed to Algorithm 6 in which FISTA is applied on the descent steps in  $\mu$  and  $\nu$  at the same time.

Finally, the last of the distinct differences, whose reason was explained in the remark made at the end of the previous subsection and will be explicitly shown in Section 4.5, is that Algorithm 7 is presented without the Lipschitz constant for the gradient  $\nabla g_{\text{sm}}$  as an input to the algorithm and that the step-size parameter  $\alpha$  is chosen independently of the Lipschitz constant, both of which are in contrary to Algorithm 6, in which  $L$  is an input and in which  $\alpha$  is chosen  $\alpha = 1/L$  (where  $L$  is the Lipschitz constant for the gradient of the corresponding maximized (dual) objective,



i.e., for  $\nabla\psi$ ). It should be noted, however, again as its reason was explained in the same remark and will be reminded of in Section 4.5, we won't actually make use of this Lipschitz constant  $L$  when implementing Algorithm 6 in Section 4.5 and will set the value of  $\alpha$  by trial and error, i.e., independently of the Lipschitz constant  $L$ . From this regard the two algorithms will be bearing a similarity in practical implementation as opposed to the difference they now bear in their theoretical forms.

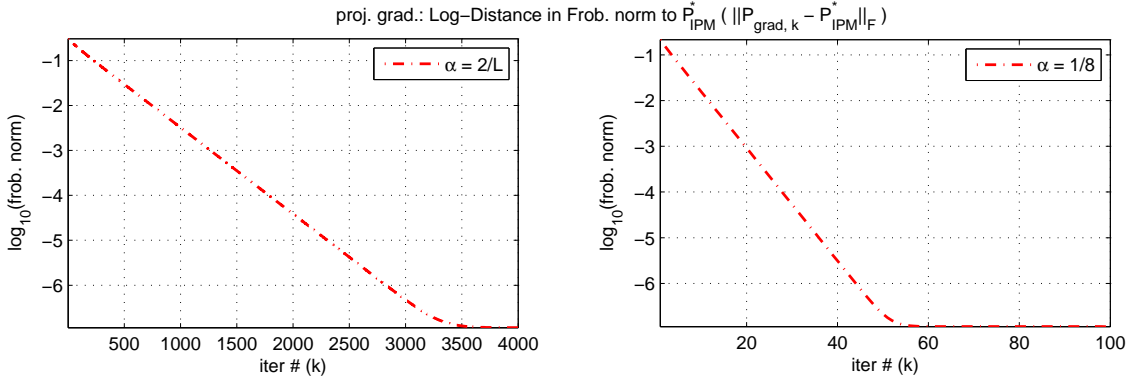
As there are between Algorithms 5 and 2, there are of course many similarities also between Algorithms 7 and 6, most important of which is the same with the one between Algorithms 5 and 2 and which was again referred to the similarity between Algorithms 4 and 1 explained previously in Subsection 4.3.2. As a summary, Algorithms 7 and 6 have similar complexity due to the fact that the main complexity of both algorithms still lies in the evaluation of the SVD needed for projection operation done onto the positive semidefinite cone once in each iteration in both algorithms.

## 4.5 Experiments and Results

In this section we present a detailed experimental analysis demonstrating the computational benefits offered by the algorithms we developed in this chapter. Throughout this analysis, we use IPM as a benchmark via the SDP optimization package SDPT3. For the solution of IPM we use an error tolerance of  $10^{-7}$  so that IPM stops iterating when both the relative gap and the infeasibilities are less than this tolerance. If we were to set the error tolerance lower, of course IPM would iterate more and converge to a finer point, but we deem the level of accuracy we achieve with this error tolerance satisfactory for any practical purpose.

In the experiments of this section we generate a number of samples from a known smooth monotone covariance  $P$  and use them to estimate  $P$ , just like we did in Subsection 3.3.2. It should be again noted that in practice one never has the 'true' covariance – here for each size  $N$  we took a sample covariance from ED data used in Section 3.3 and smoothed it, as a proxy for the true one.

The scalability of algorithms used for smooth monotone covariance estimation is one of the aspects we are interested in, as the size of the problem can get large, both in covariance estimation applications in general, and in a number of scenarios of interest in the context of the specific applications considered in this thesis. For example, the problem size gets larger as we use contracts with closer dates of expirations (e.g., monthly, as opposed to quarterly as considered in Section 3.3). Another scenario involving a larger problem size emerges when we consider combinations of several products, e.g., interest rates in EU, UK, Japan, and US, together. In this



**Figure 4.1.** Convergence characteristics ( $\|P_{\text{grad},k} - P_{\text{IPM}}^*\|_F$  at each iteration  $k$ ) of Algorithm 1 when  $N = 15$  for step sizes (a)  $\alpha = 2/L = 2/896$  (b)  $\alpha = 1/8$ .

case the number of variables grows with the number of countries (although the covariance structure will change when several curvatures are modeled jointly, similar computational approaches would still be of interest).

The structure of the rest of the section is as follows: We start with presenting the analysis of the algorithms we developed for the monotone problem, i.e., Algorithms 1 and 6 in Subsection 4.5.1. We then present the same analysis for the smooth and monotone problem, i.e., Algorithms 4 and 7 in Subsection 4.5.2.

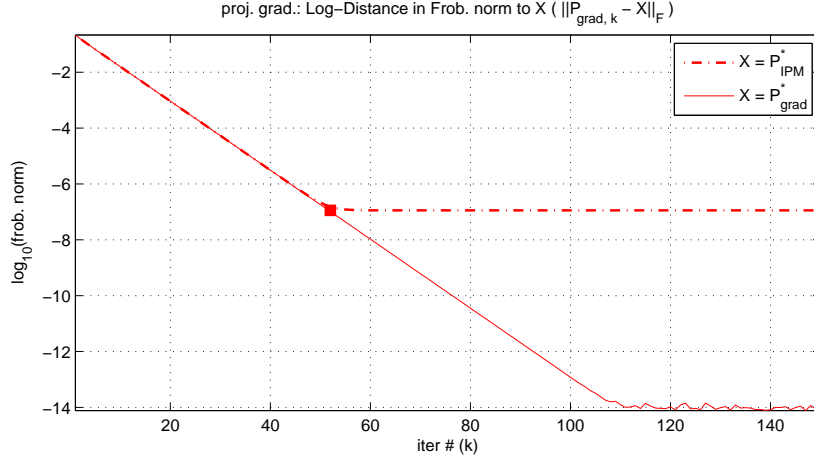
**Remark.** In order to keep parallelism with Algorithms 1 and 4, throughout the section we will be using the versions of Algorithms 6 and 7 without additional constraints.

### 4.5.1 The Monotone Problem

Now we focus on the dual projected gradient algorithm for the monotone problem (Algorithm 1) and its FISTA adaptation, the dual FISTA (Algorithm 6). We first show their individual behaviors and then proceed to comparing both with each other and with IPM.

#### Individual Behaviors of the Algorithms

We start with the simplest algorithm, Algorithm 1, the first figure regarding which we give in Figure 4.1. To construct this figure, we first find the solution via IPM, which we call  $P_{\text{IPM}}^*$ . Then, in this figure we present the typical characteristics of this algorithm for convergence to  $P_{\text{IPM}}^*$  when  $N = 15$  for different step-size parameters. In (a) we plot the convergence for step size  $\alpha = 2/L$ , which is the constant step size that guarantees convergence as mentioned in Sections 4.1 and 4.2, and where  $L$  is the Lipschitz constant for gradient  $\nabla\psi$  of the simplified dual objective  $\psi$  in (4.6), i.e., the mapping from  $\mu$  to  $\partial\psi/\partial\mu$ , and it is equal to  $L = 4(N+1)(N-1) = 896$

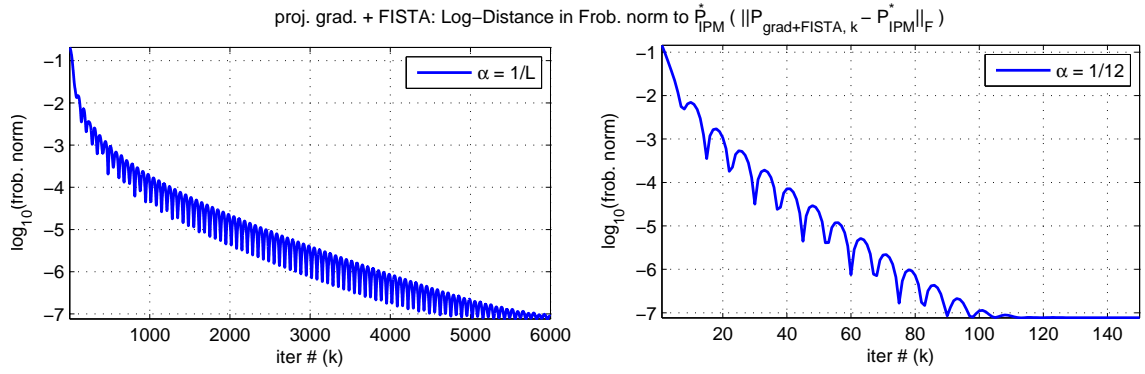


**Figure 4.2.** Characteristics of Algorithm 1 when  $N = 15$  for convergence to  $P_{\text{IPM}}^*$  and  $P_{\text{grad}}^*$  ( $\|P_{\text{grad},k} - P_{\text{IPM}}^*\|_F$  and  $\|P_{\text{grad},k} - P_{\text{grad}}^*\|_F$  at each iteration  $k$ ).

as calculated in Section 4.2. However, as mentioned in that section, this step size is too small and yields a slow convergence rate. Instead, we choose the largest value for  $\alpha$  by trial and error so that it is sufficiently small not to prevent convergence in any trial. This value for  $N = 15$  is  $\alpha = 1/8$ , and we plot the convergence for this step size in (b). As it is seen in (b), choosing this step-size parameter yields a much higher convergence rate, over 60-fold in this instance. Therefore, we will continue to choose step sizes via this method.

In the introduction of this section we mentioned that the implemented IPM stops iterating after some point because of its selected error tolerance, which yields a level of accuracy we deem satisfactory for any practical purpose. Although the optimal points for both IPM and Algorithm 1 are exactly the same, for this reason the implemented Algorithm 1 keeps descending even after reaching the proximity of  $P_{\text{IPM}}^*$  and converges to a slightly finer solution, which we will call  $P_{\text{grad}}^*$  and which is very close to (and practically indistinguishable than)  $P_{\text{IPM}}^*$ . This phenomenon is shown in Figure 4.2. (Actually the algorithm would never stop descending if we ran it on a theoretical infinite precision arithmetic computer and would converge to a even finer solution, but it practically converges to a point due to finite-precision arithmetic by which we are limited). Now we can make the definition for convergence of the algorithm to  $P_{\text{IPM}}^*$ : we say that the algorithm has converged to  $P_{\text{IPM}}^*$  at iteration  $k$  when  $\|P_{\text{grad},k} - P_{\text{IPM}}^*\|_F$  is smaller than  $\|P_{\text{grad}}^* - P_{\text{IPM}}^*\|_F$ , implying that  $P_{\text{grad},k}$  is as close to  $P_{\text{IPM}}^*$  at least as  $P_{\text{grad}}^*$  is (i.e., that  $P_{\text{grad},k}$  is practically indistinguishable than both  $P_{\text{IPM}}^*$ ), and we mark this iteration  $k$  by the square box in the figure.

Now we proceed to Algorithm 6. In Figure 4.3, we present the typical characteristics of this algorithm for convergence to  $P_{\text{IPM}}^*$  when  $N = 15$ , again for different step-size parameters. In (a) we plot the convergence for step size  $\alpha = 1/L$  (instead

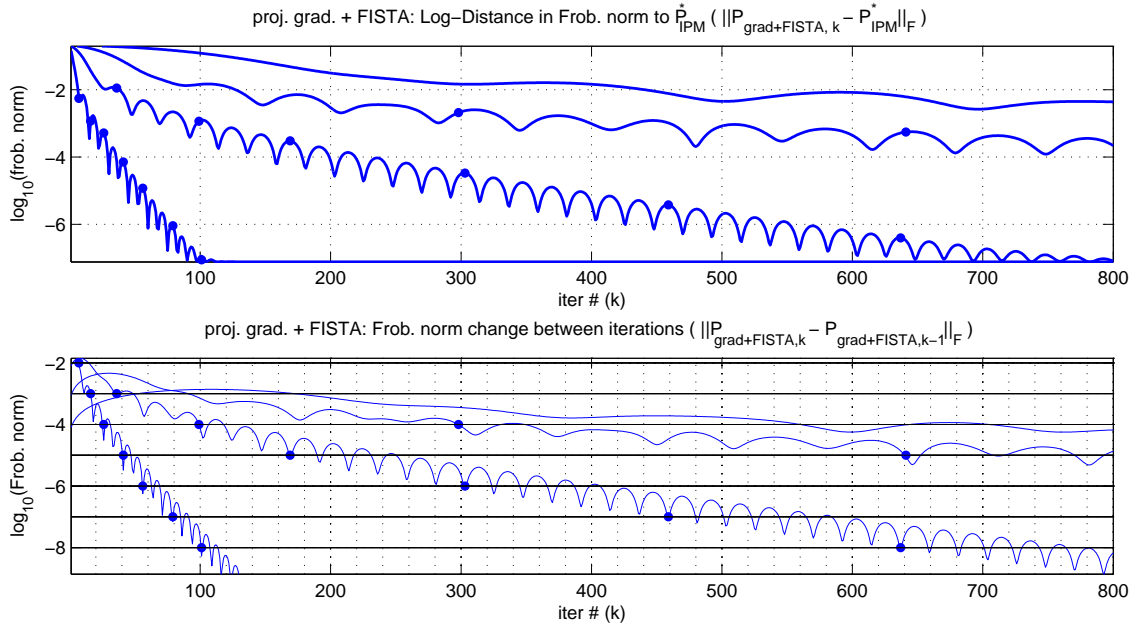


**Figure 4.3.** Convergence characteristics ( $\|P_{\text{grad+FISTA},k} - P_{\text{IPM}}^*\|_F$  at each iteration  $k$ ) of Algorithm 6 when  $N = 15$  for step sizes (a)  $\alpha = 1/L = 1/896$  (b)  $\alpha = 1/12$ .

of  $2/L$ ), which is the constant step size FISTA guarantees convergence as mentioned in Subsection 4.4.1. However, as mentioned in Subsection 4.4.2, this step size is also too small, and again we can choose the largest value for  $\alpha$  by trial and error so that it is sufficiently small not to prevent convergence in any trial. This value for  $N = 15$  is  $\alpha = 1/12$ , for which we plot the convergence behavior in (b). This step-size parameter again yields a much higher convergence rate, over 50-fold in this instance. We will also continue to choose step sizes for this algorithm via this method.

We typically observe oscillations in the descent of Algorithm 6. These downward oscillations mean that, in spite of the general descent trend of the algorithm, the covariance matrix iterates  $P_{\text{grad+FISTA},k}$  keep switching between getting closer to  $P_{\text{IPM}}^*$  and getting away from it, the latter movement in relatively lesser amount. This behavior is in contrary not only to the consistent descent behavior of Algorithm 1, but also to our expectations on theoretical considerations of FISTA, and therefore has surprised us. Moreover, although it may be thought that these oscillations are a symptom of a descent process that is too fast and that they can ameliorated by using a smaller step-size parameter, we can see that the same behavior is valid for all step-size parameters in Figure 4.4 (a), where we plot the convergence characteristics for 4 different step-size parameters, each at a different order of magnitude:  $1/12$ ,  $1/100$ ,  $1/896$ , and  $1/10000$ . We are still investigating the reason of the occurrence of these undesired oscillations in continuing work.

Nevertheless, however, although one may think that this oscillatory behavior would cause the potential risk of being near the top of an oscillation (i.e., far from  $P_{\text{IPM}}^*$ ) at the final iteration without knowing it, this situation may be prevented. For that purpose we plotted in (b) the norm change between the covariance matrix iterates. These norm change plots behave exactly opposite as their convergence plot counterparts in (a), i.e., they make a dip when there is a peak in (a), and vice versa. In (b) we mark the iterations at which norm changes hit an order of magnitude with



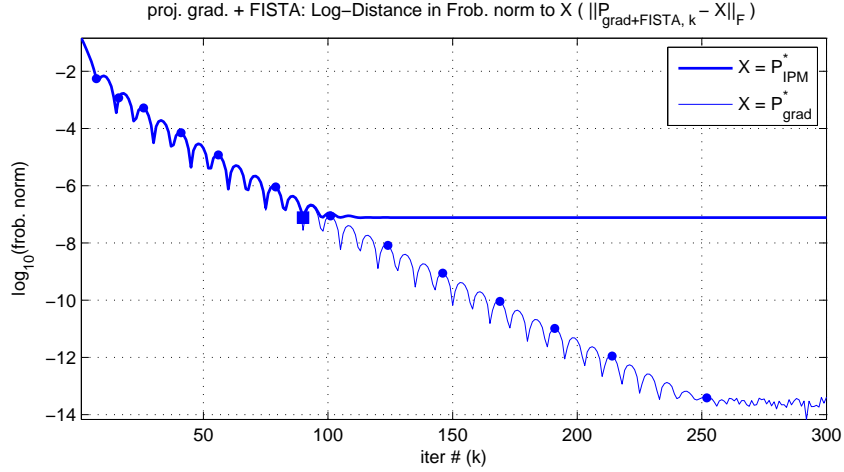
**Figure 4.4.** When  $N = 15$  and for step sizes (for curves from left to right)  $\alpha = 1/12, 1/100, 1/896, 1/1000$ , and  $1/10000$ , characteristics of Algorithm 6 regarding (a) convergence (b) norm change between the covariance matrix iterates.

circles, and use these markings in (a) exactly at the same iterations, as a summary of (b). It can be seen that when the circle is at a dip in (b), it is at a peak in (a). Therefore, since we already calculate these iterates in each iteration, we can detect the peaks in (b), enabling us to stop when we hit a dip in (a). So, effectively, we can connect the dips of curves in (a) and claim them as our convergence curves.

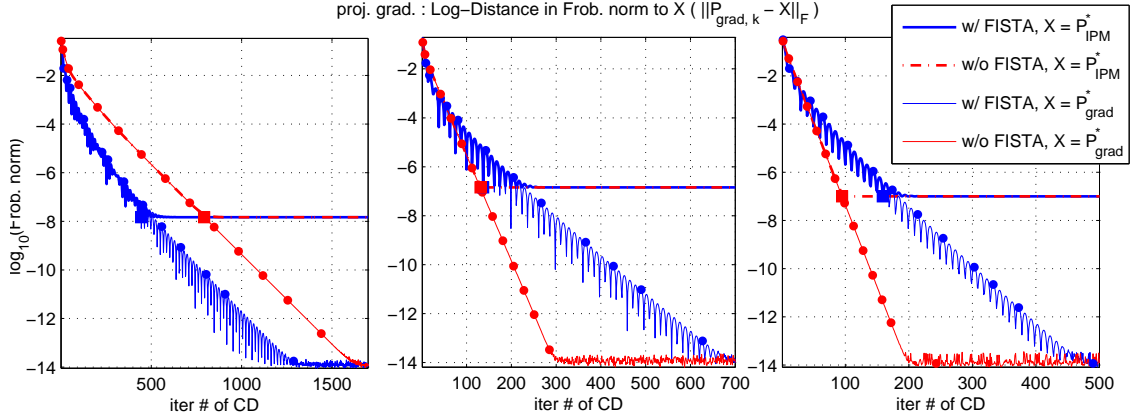
As it wasn't for Algorithm 1,  $P_{\text{IPM}}^*$  is also not the final point Algorithm 6 converges to since again the algorithm keeps descending to a slightly finer point as shown in Figure 4.5. This descent continues until the algorithm reaches the same point Algorithm 1 converges to, i.e.,  $P_{\text{grad}}^*$ . Every point we have previously made about the relation between  $P_{\text{IPM}}^*$  and  $P_{\text{grad}}^*$  during the discussion of Algorithm 1 is still valid. We also define convergence of the algorithm to  $P_{\text{IPM}}^*$  the same way we did for Algorithm 1 and mark the convergence iteration  $k$  again by the square box in the figure.

## Comparison

We now start comparing Algorithms 1 and 6 both with each other and with IPM. It is important to remind here that solving an SDP such as our problem via an IPM can become unduly computationally expensive for large covariance matrices, as it involves computing the Hessian. In Figure 4.6 we present three illustrative plots in each of which relative performance of Algorithm 6 with respect to Algorithm 1



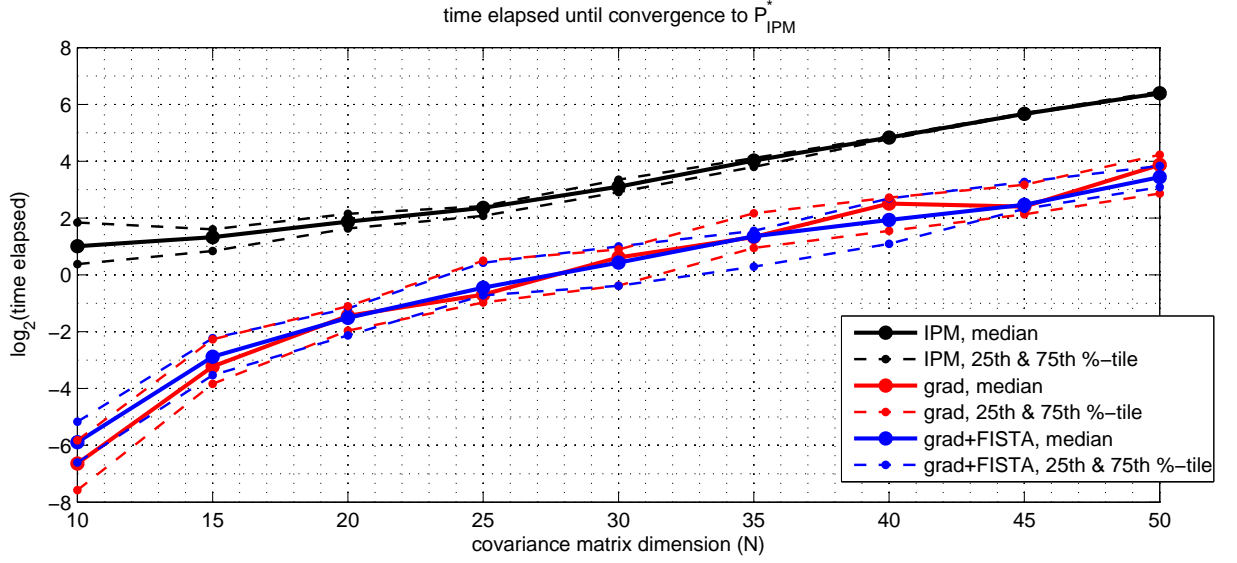
**Figure 4.5.** Characteristics of Algorithm 6 when  $N = 15$  for convergence to  $P_{\text{IPM}}^*$  and  $P_{\text{grad}}^*$  ( $\|P_{\text{grad}+\text{FISTA},k} - P_{\text{IPM}}^*\|_F$  and  $\|P_{\text{grad}+\text{FISTA},k} - P_{\text{grad}}^*\|_F$  at each iteration  $k$ ).



**Figure 4.6.** Different performances of Algorithm 6 (blue curves) with respect to Algorithm 1 (red curves) for different samples: Algorithm 6 converges to  $P_{\text{IPM}}^*$  (a) faster than (b) as fast as (c) slower than Algorithm 1.

changes with different samples obtained from the true covariance matrix. As it can be seen in (b) and (c), however, one behavior is sometimes observed: First Algorithm 6 descends faster, then Algorithm 1 takes the lead and is the first one to converge to  $P_{\text{grad}}^*$ . When this is the case, the question of which one converges to  $P_{\text{IPM}}^*$  faster, a question which we are interested in, depends on when Algorithm 1 catches up Algorithm 6. We are also interested in the question of how much Algorithm 6 is advantageous with respect to Algorithm 1 when less accuracy of the solution, e.g.,  $10^{-2}$  or  $10^{-3}$  is sufficient.

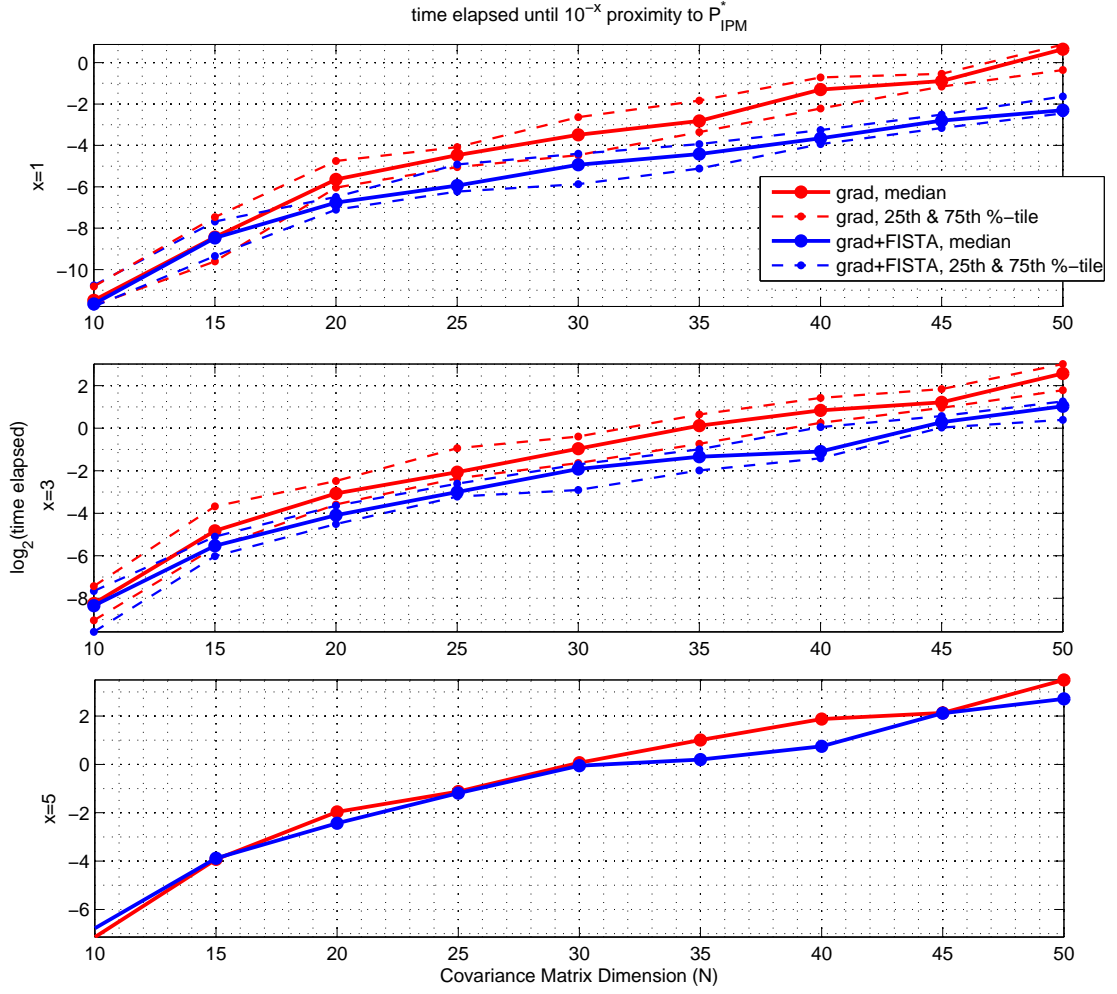
We now proceed to timing analysis and first show in Figure 4.7 the timing analysis for IPM, Algorithm 1, and Algorithm 6 regarding the time it takes to converge to  $P_{\text{IPM}}^*$  for dimensions from  $N = 10$  to  $N = 50$ . We note that we measure this timing in running time rather than in number of iterations, as the former is what matters at the end regarding convergence speed. For this purpose we repeat the experiments



**Figure 4.7.** Median and 25<sup>th</sup>-75<sup>th</sup> percentile time it takes for IPM (black curves), Algorithm 1 (red curves), and Algorithm 6 (blue curves) to converge to  $P_{\text{IPM}}^*$  for  $N$  from 10 to 50.

we have done 20 times for each  $N$ , and then use median and 25<sup>th</sup> and 75<sup>th</sup> percentile information from these experiments. We see that for any  $N$ , Algorithms 1 and 6 converge to  $P_{\text{IPM}}^*$  at least about 5 and 8 times faster than IPM, respectively.

Although Algorithm 6 seems to be a bit more advantageous for  $N = 40$  and  $50$  in Figure 4.7, to our surprise the major of the theoretical advantage of FISTA as it promises on theoretical grounds is not as evident in its practical performance, and for further work we will not only investigate it closely but also experiment with other first-order methods. Nevertheless, the realized advantage of FISTA in Algorithm 6 is not limited to that observed in Figure 4.7 and is more evident in Figure 4.8, in which we give a timing analysis of Algorithms 1 and 6 again for the same  $N$  values, but this time regarding the time it takes to reach  $10^{-x}$  proximity of  $P_{\text{IPM}}^*$  for some  $x$  values (instead of the time it takes to reach  $\|P_{\text{IPM}}^* - P_{\text{grad}}^*\|_F$ , i.e., to converge to  $P_{\text{IPM}}^*$  by our definition). Again we measure the timing in running time. Here the advantage of Algorithm 6 can be seen more clearly: Algorithm 6 reaches  $10^{-x}$  proximity of  $P_{\text{IPM}}^*$  faster than Algorithm 1 does for  $x = 1, 3$  (and also for 2 and 4, although not shown) at least 87.5% of the time and for  $x = 5$  at least 50% of the time. Moreover, on average Algorithm 6 reaches  $10^{-x}$  proximity of  $P_{\text{IPM}}^*$  for at least one value of  $N$  up to 8, 16, 15, 7, and 4 times faster than Algorithm 1 for  $x = 1, 2, 3, 4$ , and 5 respectively, the gap generally increasing for larger  $N$ .



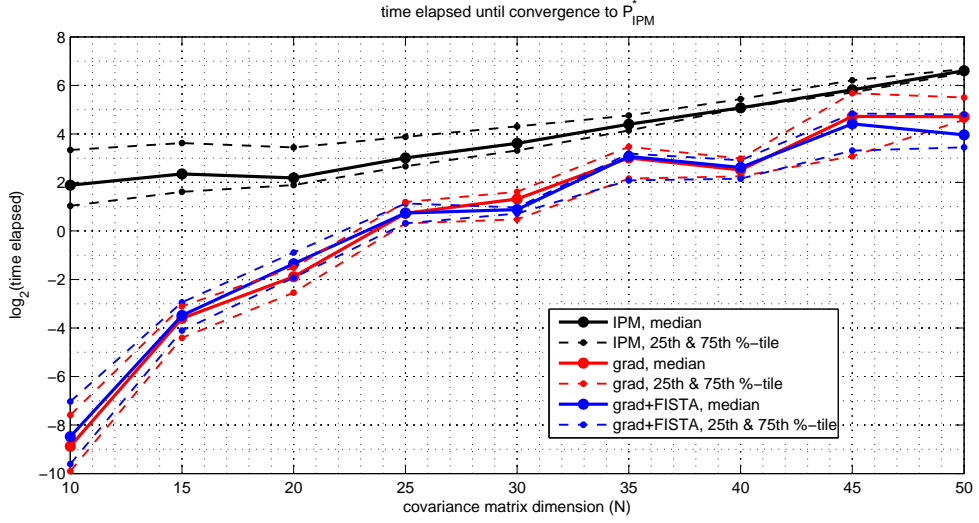
**Figure 4.8.** Median time it takes for Algorithm 1 (red curves) and Algorithm 6 (blue curves) to reach to  $10^{-x}$  proximity of  $P_{\text{IPM}}^*$  for (a)  $x = 1$  (b)  $x = 3$  (c)  $x = 5$ , including 25<sup>th</sup>-75<sup>th</sup> percentiles for (a) and (b), for  $N$  from 10 to 50.

#### 4.5.2 The Smooth and Monotone Problem

Now we focus on the dual projected gradient algorithm for the smooth and monotone problem (Algorithm 4) and its FISTA adaptation, the dual FISTA (Algorithm 7). Since when we tested we observed that their individual behaviors are the same as the algorithms analyzed in the previous subsection and that there is no change, we directly proceed to comparing the timing results of the algorithms again both with each other and with IPM.

We first show in Figure 4.9 the timing analysis for IPM, Algorithm 4 and Algorithm 7 regarding the time it takes to converge to  $P_{\text{IPM}}^*$  for dimensions from  $N = 10$  to  $N = 50$ , where we again measure this timing in running time rather than in number of iterations. For this purpose we repeat the experiments we have done 20 times for each  $N$ , and then use median and 25<sup>th</sup> and 75<sup>th</sup> percentile information

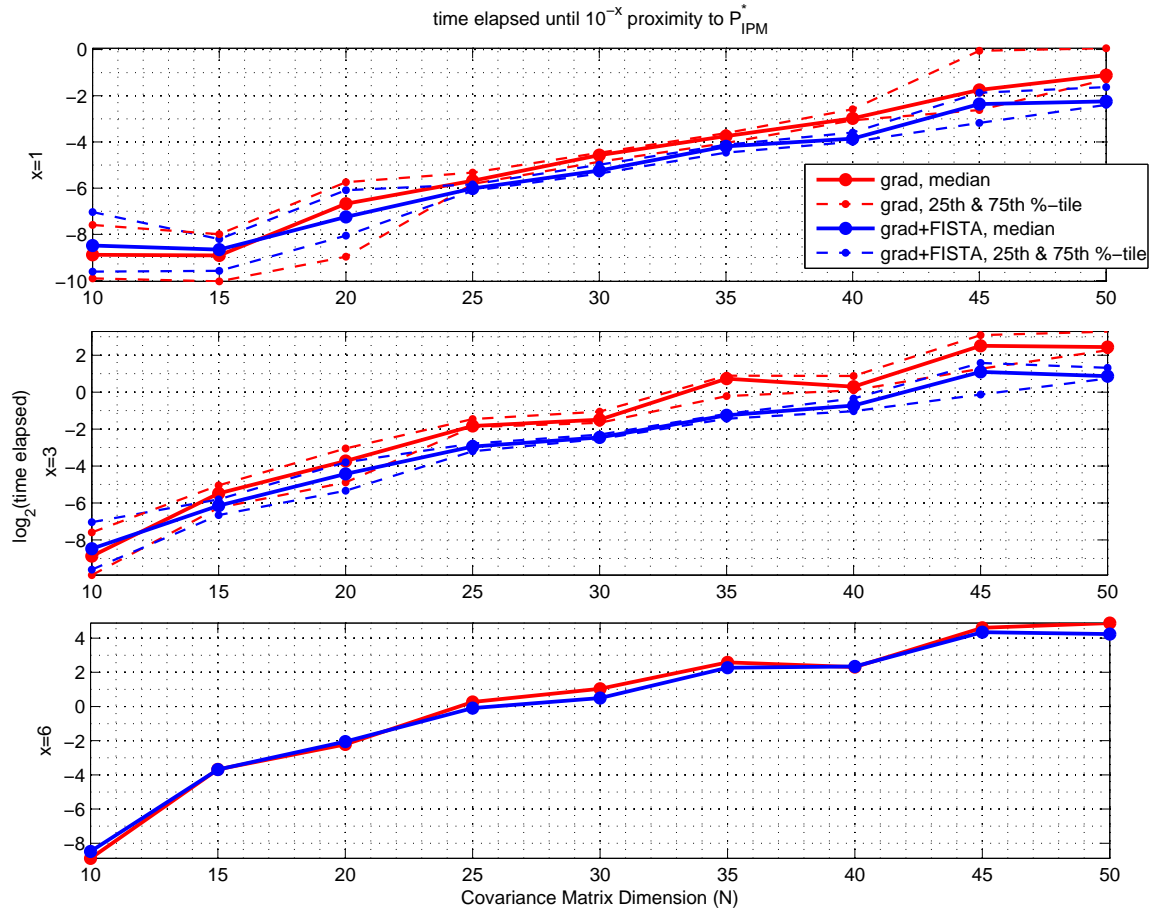




**Figure 4.9.** Smooth-monotone: median and 25<sup>th</sup>-75<sup>th</sup> percentile time it takes for IPM (black curves), Algorithm 4 (red curves), and Algorithm 7 (blue curves) to converge to  $P_{\text{IPM}}^*$  for  $N$  from 10 to 50.

from these experiments. We see that for any  $N$ , both Algorithm 4 and 7 converge to  $P_{\text{IPM}}^*$  at least 2 times faster than IPM.

Although Algorithm 7 seems to be a bit more advantageous for  $N = 30, 45$  and 50 in Figure 4.9, the major of the theoretical advantage of FISTA as it promises on theoretical grounds is again not as evident in its practical performance, a result not in the fully desired direction which will be investigated in the future work as mentioned previously. Nevertheless, the realized advantage of FISTA in Algorithm 7 is again not limited to that observed in Figure 4.9 and is more evident in Figure 4.10, in which we give a timing analysis of Algorithms 4 and 7 again for the same  $N$  values, but this time regarding the time it takes to reach  $10^{-x}$  proximity of  $P_{\text{IPM}}^*$  for some  $x$  values (instead of the time it takes to reach  $\|P_{\text{IPM}}^* - P_{\text{grad}}^*\|_F$ , i.e., to converge to  $P_{\text{IPM}}^*$  by our definition). Here the advantage of Algorithm 7 can be seen more clearly: Algorithm 7 reaches  $10^{-x}$  proximity of  $P_{\text{IPM}}^*$  faster than Algorithm 4 does for  $x = 1, 3$  (and also for 2 and 4, although not shown) at least 87.5% of the time, for  $x = 5$  at least 75% of the time, and for  $x = 6$  at least 50% of the time. Moreover, on average Algorithm 7 reaches to  $10^{-x}$  proximity of  $P_{\text{IPM}}^*$  for at least one value of  $N$  up to 6, 8, 16, 16, 8 and 1.5 times faster than Algorithm 4 for  $x = 1, 2, 3, 4, 5$ , and 6 respectively, the gap generally enlarging for larger  $N$ .



**Figure 4.10.** Smooth-monotone: median time it takes for Algorithm 4 (red curves) and Algorithm 7 (blue curves) to reach to  $10^{-x}$  proximity of  $P_{\text{IPM}}^*$  for (a)  $x = 1$  (b)  $x = 3$  (c)  $x = 6$ , including 25<sup>th</sup>-75<sup>th</sup> percentiles for (a) and (b), for  $N$  from 10 to 50.

# Chapter 5

## Conclusion

In this thesis the problem of interest is covariance matrix estimation from limited number of high dimensional i.i.d. multivariate samples when the random variables of interest have a natural spatial indexing along a low-dimensional manifold, e.g., along a line. For this problem we take as basis the smooth-monotone estimation formulation that allows all the elements of the covariance matrix to be treated as separate parameters, but requires the covariance function to be smooth and monotone with respect to this indexing. The primary aim of the thesis is to develop highly efficient first-order solvers for this smooth-monotone formulation. The secondary aim is to present extensive simulations of (1) the developed first order solvers, which are based on this formulation, regarding their computational benefits and of (2) the smooth-monotone covariance estimation formulation regarding its accuracy.

In this chapter we first summarize the thesis and the contributions in Section 5.1. In Section 5.2 we discuss several extensions to the ideas presented in the thesis, with a number of suggestions for further research.

### 5.1 Summary of the Thesis and of the Contributions

After providing a background in Chapter 2, we proceeded to Chapter 3, which contained the formulation we used as basis for covariance estimation. In that chapter, after motivating the use of this formulation and posing the estimation problem in a convex-optimization framework, we re-presented the solution of the resulting semidefinite-programming problem by an interior-point method (We also mentioned that solving an SDP this way can become unduly computationally expensive for large covariance matrices, as it involves computing the Hessian.). Then we started to make our own contributions one-by-one:

1. In Section 3.3, we made our first contribution by demonstrating the application of our approach on a number of examples with limited, missing and

asynchronous data, and showing that it has the potential to provide more accurate covariance matrix estimates than existing methods, and exhibits a desirable eigenvalue-spectrum correction effect.

2. We then proceeded to Chapter 4, in which after a quick revisit to the original gradient projection method developed in [26] we made our main contribution through Sections 4.2 - 4.4 by developing optimal first-order methods for solving our optimization problem. In our derivation we first adapted the projected gradient method of [26] in Sections 4.2 and 4.3 and accelerated them following the optimal first order ideas of [25] in Section 4.4. To be specific, we first described a dual first-order method for the special case of our problem which contains only monotonicity constraints in Section 4.2 for pedagogical reasons and then a dual projected coordinate descent solution for our smooth and monotone problem in Section 4.3.
3. Finally, we presented our final contribution in Section 4.5 as a detailed experimental analysis demonstrating the computational benefits offered by the algorithm we developed in Chapter 4.

## 5.2 Future Work

We can divide potential future work to three categories: application, analysis, and formulation.

### 5.2.1 Application

The first future application we can study is a direct extension of the one we already did. We presented in Section 3.3 the application of the smooth-monotone formulation on term-structure modeling, where in Section 3.3.4 we presented a study of forecasting future correlation coefficient matrices over several years of historical data of ED prices. We can take this one step further and forecast future covariance matrices instead of just correlation coefficient matrices, by using GARCH (Generalized Autoregressive Conditional Heteroskedasticity) modeling [41] to predict the diagonal of variances and fusing it with our smooth-monotone estimate of the correlation structure. Then we can directly use the predicted covariance matrices to see the performance improvement in the allocation of the assets in the mentioned portfolio selection methods, such as Markowitz portfolios.

Other potential applications include extension of the smooth-monotone framework to 2D regular grids, for example in modeling volatility surfaces [40].

### 5.2.2 Analysis

The most important future work in this category is the investigation of FISTA further in an attempt to find out the reason why the major of the theoretical advantage of FISTA as it promises on theoretical grounds is not as evident in practical performances of Algorithms 6 and 7 as shown and discussed in Section 4.5. An important portion of this investigation is the study of the possible reasons of the undesired oscillations observed when these algorithms are implemented, in contrary to our expectations on theoretical considerations of FISTA.

As a separate future work, a more extensive analysis of the algorithms we have developed can be sought. Two of possible investigations of this kind could be regarding the theoretical characterization of the convergence rates and of the number of samples required for convergence, where in the latter convergence is in terms of estimation accuracy as opposed to optimization accuracy (i.e., converging to the true minimizer) as meant in the former. Other possible guarantees of numerical performance can be investigated as well.

Finally, the performance of our algorithms can be compared to alternative first-order methods as well in addition to IPM, such as NESTA and its variations [39], or those studied in [44-50].

### 5.2.3 Formulation

The first formulation-wise improvement could be regarding the first order ideas implemented. The most basic step in this direction would be instead of adapting FISTA with constant step size to our Algorithms 1 and 4, adapting FISTA with backtracking which was again described in [25]. This version of FISTA, which uses adaptive step size instead of a constant step size, may provide faster convergence by initializing a larger step size and increasing it when necessary. A larger step would be using other first-order alternatives to FISTA altogether, such as NESTA and its variations [39], or [44-50] as mentioned above.

A more fundamental attempt could be using alternative optimization methods instead of the gradient projection method of [26] to solve our optimization problem. One example to these alternative methods is [51], which deals with the same problem as [26].

A third possible improvement could be achieved by choosing a different error metric  $D(P, \hat{P})$  for our formulation as we mentioned in Section 3.2, for example we can also use Kullback-Leibler (KL) divergence, which for two-zero mean Gaussian distributions with covariances  $P$  and  $Q$  is defined as (see (3.4))

$$D(P||Q) = \frac{1}{2}[\log(\det QP^{-1})] + \text{tr}(QP^{-1}) - N].$$

Of course, using such an error metric would change the optimization method we need to use since the gradient projection method of [26] is applicable only when the error metric is Frobenius norm.

# Appendix A

## Mathematical Preliminaries

In this appendix we provide some mathematical preliminaries which will be of use in the thesis. The material in this section is borrowed from [33], to which the reader is referred for a more thorough understanding.

### A.1 Convex Sets and Cones, and Relation to Positive Semidefiniteness and Generalized Inequalities

#### Convex Sets

A set  $C$  is a *convex* if the line segment between any two points in  $C$  lies in  $C$ , i.e., if for any  $x_1, x_2 \in C$  and any  $\theta$  with  $0 \leq \theta \leq 1$ , we have

$$\theta x_1 + (1 - \theta)x_2 \in C.$$

We call a point of the form  $\theta_1 x_1 + \dots + \theta_k x_k$ , where  $\theta_1 + \dots + \theta_k = 1$  and  $\theta_i \geq 0, i = 1, \dots, k$ , a convex combination of the points  $x_1, \dots, x_k$ . A set is convex if and only if it contains every convex combination of its points. A convex combination of points can be thought of as a mixture or weighted average of the points, with  $\theta_i$  the fraction of  $x_i$  in the mixture.

#### Cones and Convex Cones

A set  $C$  is called a *cone* if for every  $x \in C$  and  $\theta \geq 0$  we have  $\theta x \in C$ . A set  $C$  is a *convex cone* if it is convex and a cone, which means that for any  $x_1, x_2 \in C$  and  $\theta_1, \theta_2 \geq 0$ , we have

$$\theta_1 x_1 + \theta_2 x_2 \in C.$$

A point of the form  $\theta_1 x_1 + \dots + \theta_k x_k$  with  $\theta_1, \dots, \theta_k \geq 0$  is called a *conic combination* of  $x_1, \dots, x_k$ . If  $x_i$  are in a convex cone  $C$ , then every conic combination of  $x_i$  is in  $C$ .

Conversely, a set  $C$  is a convex cone if and only if it contains all conic combinations of its elements.

### The Positive Semidefinite Cone

We use the notation  $\mathbf{S}^n$  to denote the set symmetric  $n \times n$  matrices,

$$\mathbf{S}^n = \{X \in \mathbb{R}^{n \times n} \mid X = X^T\}$$

which is a vector space with dimension  $n(n+1)/2$ . We use the notation  $\mathbf{S}_+^n$  to denote the set of symmetric positive semidefinite matrices:

$$\mathbf{S}_+^n = \{X \in \mathbf{S}^n \mid X \succeq 0\},$$

where the *matrix inequality*  $X \succeq 0$  means that  $X$  is *positive semidefinite*, i.e.,  $z^T X z \geq 0$  for all non-zero vectors  $z$  with real entries ( $z \in \mathbb{R}^n$ ). If moreover  $z^T X z > 0$  for all such  $z$ , then it means that  $X$  is *positive definite* and we denote it by  $X \succ 0$ . We use the notation  $\mathbf{S}_{++}^n$  to denote the set of symmetric positive definite matrices:

$$\mathbf{S}_{++}^n = \{X \in \mathbf{S}^n \mid X \succ 0\}.$$

(This notation is meant to be analogous to  $\mathbb{R}_+$ , which denotes the nonnegative reals, and  $\mathbb{R}_{++}$ , which denotes the positive reals.)

The set  $\mathbf{S}_+^n$  is a convex cone: if  $\theta_1, \theta_2 \geq 0$  and  $A, B \in \mathbf{S}_+^n$ , then  $\theta_1 A + \theta_2 B \in \mathbf{S}_+^n$ . This can be seen directly from the definition of positive semidefiniteness: for any  $x \in \mathbb{R}^n$ , we have

$$x^T(\theta_1 A + \theta_2 B)x = \theta_1 x^T A x + \theta_2 x^T B x \geq 0,$$

if  $A \succeq 0, B \succeq 0$  and  $\theta_1, \theta_2 \geq 0$ .

### Generalized inequalities and Matrix Inequality

A cone  $K \subseteq \mathbb{R}^n$  is called a *proper cone* if it is at the same time (1) convex, (2) closed, (3) *solid*, which means it has nonempty interior, and (4) *pointed*, which means that it contains no line.

A proper cone  $K$  can be used to define a *generalized inequality*, which is a partial ordering on  $\mathbb{R}^n$  that has many of the properties of the standard ordering on  $\mathbb{R}$  (refer to p. 44 in [33] for a full list of these properties). We associate with the proper cone  $K$  the partial ordering on  $\mathbb{R}^n$  defined by

$$x \succeq_K y \iff x - y \in K.$$



We also write  $y \preceq_K x$  for  $x \succeq_K y$ . Similarly, we define an associated strict partial ordering by

$$x \succ_K y \iff x - y \in \text{int}K,$$

and write  $y \prec_K x$  for  $x \succ_K y$ . (To distinguish the generalized inequality  $\succeq_K$  from the strict generalized inequality, we sometimes refer to  $\succeq_K$  as the nonstrict generalized inequality.)

When  $K = \mathbb{R}_+$ , the partial ordering  $\succeq_K$  is the usual ordering  $\geq$  on  $\mathbb{R}$ , and the strict partial ordering  $\succ$  is the same as the usual strict ordering  $>$  on  $\mathbb{R}$ . So generalized inequalities include as a special case ordinary (nonstrict and strict) inequality in  $\mathbb{R}$ .

The positive semidefinite cone  $\mathbf{S}_+^n$  is a proper cone in  $\mathbf{S}^n$ . The associated generalized inequality  $\succeq_K$  is the usual matrix inequality:  $X \succeq_K Y$  means  $X - Y$  is positive semidefinite. The interior of  $\mathbf{S}_+^n$  (in  $\mathbf{S}^n$ ) consists of the positive definite matrices, so the strict generalized inequality also agrees with the usual strict inequality between symmetric matrices:  $X \succ_K Y$  means  $X - Y$  is positive definite. Here, too, the partial ordering arises so frequently that we drop the subscript: for symmetric matrices we write simply  $X \succeq Y$  or  $X \succ Y$ . It is understood that the generalized inequalities are with respect to the positive semidefinite cone.

### Projection onto the Positive Semidefinite Cone

The material in this part is borrowed from [26]. The positive and negative semidefinite parts of a  $N \times N$  symmetric matrix  $X$ , denoted by  $X_+$  and  $X_-$ , respectively, are defined implicitly by the conditions

$$X = X_+ - X_-, \quad X_+ = X_+^T \succeq 0, \quad X_- = X_-^T \succeq 0, \quad X_+ X_- = 0.$$

The positive semidefinite part  $X_+$  is the projection of  $X$  onto the positive semidefinite cone, i.e., we have

$$\|X - X_+\|_F = \|X_-\|_F \leq \|X - Z\|_F. \quad (\text{A.1})$$

for any positive semidefinite  $Z$ . In a similar way,  $\|X + Z\|_F$  is minimized, over all positive semidefinite matrices  $Z$ , by the choice of  $Z = X_-$  (see, e.g., [33, Section 8.1.1]).

We can express the positive and negative semidefinite parts explicitly as

$$X_+ = \sum_{\lambda_i > 0} \lambda_i q_i q_i^T, \quad X_- = \sum_{\lambda_i < 0} \lambda_i q_i q_i^T, \quad (\text{A.2})$$

where  $X = \sum_{i=1}^N \lambda_i q_i q_i^T$  is an eigendecomposition of  $X$ , i.e.,  $q_1, \dots, q_N$  is a set of orthonormal eigenvectors of  $X$  with corresponding eigenvalues  $\lambda_1, \dots, \lambda_N$ .

## A.2 Convex Optimization

### Convex Functions

A function  $f : \mathbb{R}^n \rightarrow \mathbb{R}$  is *convex* if  $\mathbf{dom} f$  is a convex set and if for all  $x, y \in \mathbf{dom} f$ , and  $\theta$  with  $0 \leq \theta \leq 1$ , we have

$$f(\theta x + (1 - \theta)y) \leq \theta f(x) + (1 - \theta)f(y). \quad (\text{A.3})$$

A function  $f$  is *strictly convex* if strict inequality holds in (A.3) whenever  $x \neq y$  and  $0 < \theta < 1$ . We say  $f$  is *concave* if  $-f$  is convex, and *strictly concave* if  $-f$  is strictly convex.

Suppose  $K \subseteq \mathbb{R}^m$  is a proper cone with associated generalized inequality  $\preceq_K$ . We say  $f : \mathbb{R}^m \rightarrow \mathbb{R}^n$  is *K-convex* if it satisfies

$$f(\theta x + (1 - \theta)y) \preceq_K \theta f(x) + (1 - \theta)f(y) \quad (\text{A.4})$$

for all  $x, y$ , and  $0 \leq \theta \leq 1$ , and *strictly K-convex* if for all  $x \neq y$  and  $0 < \theta < 1$  it satisfies

$$f(\theta x + (1 - \theta)y) \prec_K \theta f(x) + (1 - \theta)f(y). \quad (\text{A.5})$$

These definitions reduce to ordinary convexity and strict convexity when  $m = 1$  (and  $K = \mathbb{R}_+$ ).

The  $\alpha$ -*sublevel set* of a function  $f : \mathbb{R}^n \rightarrow \mathbb{R}$  is defined as

$$C_\alpha = \{x \in \mathbf{dom} f \mid f(x) \leq \alpha\},$$

Sublevel sets of a convex function are convex, for any value of  $\alpha$ .

### Convex Optimization Problem

The simplest form of a convex optimization problem is

$$\begin{aligned} & \text{minimize} && f_0(x) \\ & \text{subject to} && f_i(x) \leq 0, \quad \text{for } i = 1, \dots, m, \end{aligned} \quad (\text{A.6})$$

where the functions  $f_0, \dots, f_m : \mathbb{R}^n \rightarrow \mathbb{R}$  are convex.

It is easy to modify (A.6) to include equality constraints. Suppose that we want to add the equality constraint  $h(x) = 0$  where again  $h : \mathbb{R}^n \rightarrow \mathbb{R}$ . We may include this constraint in (A.6) via adding both  $h(x) \leq 0$  and  $-h(x) \leq 0$  to the constraint set. To be able to do this, however, both  $h(x)$  and  $-h(x)$  have to be convex, meaning that  $h(x)$  has to be linear, or affine, i.e., has to be expressible in the form

$h(x) = a^T x - b$  where  $a, b$  are column vectors. Therefore, we can now express a convex optimization in its standard form

$$\begin{aligned} & \text{minimize} && f_0(x) \\ & \text{subject to} && f_i(x) \leq 0, \quad \text{for } i = 1, \dots, m, \\ & && a_j^T x = b_j, \quad \text{for } j = 1, \dots, p, \end{aligned} \tag{A.7}$$

where the functions  $f_0, \dots, f_m : \mathbb{R}^n \rightarrow \mathbb{R}$  are again convex and  $a_j \in \mathbb{R}^n, b_j \in \mathbb{R}$  for all  $j = 1, \dots, p$ .

Now, suppose that we now want use matrix representation  $X$  instead of column vector representation  $x$ . We can again easily modify (A.7) for this representation. In this case, a convex optimization problem can be expressed as

$$\begin{aligned} & \text{minimize} && f_0(X) \\ & \text{subject to} && f_i(X) \leq 0, \quad \text{for } i = 1, \dots, m, \\ & && \mathbf{Tr} A_j X = b_j, \quad \text{for } j = 1, \dots, p, \end{aligned} \tag{A.8}$$

where the functions  $f_0, \dots, f_m : \mathbb{R}^{M \times N} \rightarrow \mathbb{R}$  are again convex,  $A_j \in \mathbb{R}^{N \times M}$  for all  $j = 1, \dots, p$ , and  $\mathbf{Tr}$  denotes the trace of a matrix, i.e., if  $U$  is a  $N \times N$  matrix, then

$$\mathbf{Tr} U = \sum_{i=1}^N [U]_{(i,i)}.$$

**Remark.** If the objective  $f_0$  of convex optimization problem (A.6) (or equivalently of (A.7) or (A.8)) has bounded sublevel sets and, in addition, is strictly convex, then the feasibility of convex optimization problem is a sufficient condition for it to have a unique solution  $x^*$  (or  $X^*$ ).

## Generalized Inequality Constraints and Semidefinite Programming

One very useful generalization of the standard form convex optimization problem (A.7) is obtained by allowing the inequality constraint functions to be vector valued, and using generalized inequalities in the constraints:

$$\begin{aligned} & \text{minimize} && f_0(x) \\ & \text{subject to} && f_i(x) \prec_{K_i} 0, \quad \text{for } i = 1, \dots, m, \\ & && Ax = b, \end{aligned} \tag{A.9}$$

where  $f_0 : \mathbb{R}^n \rightarrow \mathbb{R}, K_i \subseteq \mathbb{R}^{k_i}$  are proper cones, and  $f_i : \mathbb{R}^n \rightarrow \mathbb{R}^{k_i}$  are  $K_i$ -convex. We refer to this problem as a (standard) form *convex optimization problem with generalized inequality constraints*. Problem (A.7) is a special case with  $K_i = \mathbb{R}_+$  for all  $i = 1, \dots, m$ .

The type of problem that will be most relevant to us under this category is *Semidefinite Programming*, which is subcategory for  $K_i = \mathbf{S}_+^N$ . A *standard form SDP* has linear equality constraints, and a (matrix) nonnegativity constraint on the variables  $X \in \mathbf{S}^N$ :

$$\begin{aligned} & \text{minimize} && \mathbf{Tr} \, CX \\ & \text{subject to} && X \succeq 0, \\ & && \mathbf{Tr} \, A_i X = b_i, \quad \text{for } i = 1, \dots, p, \end{aligned} \tag{A.10}$$

where  $C, A_1, \dots, A_p \in \mathbf{S}^n$  and  $b_i \in \mathbb{R}$  for all  $i = 1, \dots, p$ . The version of this problem with additional linear inequality constraints, as given in the following, is also a SDP:

$$\begin{aligned} & \text{minimize} && \mathbf{Tr} \, CX \\ & \text{subject to} && X \succeq 0, \\ & && \mathbf{Tr} \, A_i X = b_i, \quad \text{for } i = 1, \dots, p, \\ & && \mathbf{Tr} \, C_j X \leq d_j, \quad \text{for } j = 1, \dots, m, \end{aligned} \tag{A.11}$$

where  $C, A_1, \dots, A_p, C_1, \dots, C_m \in \mathbf{S}^n$  and  $b_1, \dots, b_p, d_1, \dots, d_m \in \mathbb{R}$ .

### A.3 Duality

In this section we cover Lagrangian duality, which plays a central role in convex optimization and which will be of specific importance in the thesis.

#### The Lagrangian

We consider an (not necessarily convex) optimization problem in the standard form:

$$\begin{aligned} & \text{minimize} && f_0(x) \\ & \text{subject to} && f_i(x) \leq 0, \quad \text{for } i = 1, \dots, m, \\ & && h_i(x) = 0, \quad \text{for } i = 1, \dots, p, \end{aligned} \tag{A.12}$$

with variable  $x \in \mathbb{R}^n$ . We assume its domain  $D = (\cap_{i=0}^m \mathbf{dom} f_i) \cap (\cap_{i=1}^p \mathbf{dom} h_i)$  is nonempty, and denote the optimal value of (A.12) by  $p^*$ .

The basic idea in Lagrangian duality is to take the constraints in (A.12) into account by augmenting the objective function with a weighted sum of the constraint functions. We define the *Lagrangian*  $L : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^p \rightarrow \mathbb{R}$  associated with the problem (A.12) as

$$L(x, \mu, \nu) = f_0(x) + \sum_{i=1}^m \mu_i f_i(x) + \sum_{i=1}^p \nu_i h_i(x),$$

with  $\mathbf{dom} L = D \times \mathbb{R}^m \times \mathbb{R}^p$ . We refer to  $\mu_i$  as the *Lagrange multiplier* associated with the  $i$ th inequality constraint  $f_i(x) \leq 0$ ; similarly we refer to  $\nu_i$  as the Lagrange multiplier associated with the  $i$ th equality constraint  $h_i(x) = 0$ . The vectors  $\mu$  and  $\nu$  are called the *dual variables* or *Lagrange multiplier vectors* associated with the problem (A.12).

### The Lagrange Dual Function

We define the *Lagrange dual function* (or just *dual function*)  $g : \mathbb{R}^m \times \mathbb{R}^p \rightarrow \mathbb{R}$  as the minimum value of the Lagrangian over  $x$ : for  $\mu \in \mathbb{R}^m, \nu \in \mathbb{R}^p$ ,

$$g(\mu, \nu) = \inf_{x \in D} L(x, \mu, \nu) = \inf_{x \in D} (f_0(x) + \sum_{i=1}^m \mu_i f_i(x) + \sum_{i=1}^p \nu_i h_i(x)).$$

When the Lagrangian is unbounded below in  $x$ , the dual function takes on the value  $-\infty$ . Since the dual function is the pointwise infimum of a family of affine functions of  $(\mu, \nu)$ , it is concave, even when the problem (A.12) is not convex. The dual function yields lower bounds on the optimal value  $p^*$  of the problem (A.12): For any  $\mu \succeq 0$  and any  $\nu$  we have

$$g(\mu, \nu) \leq p^*, \tag{A.13}$$

which follows from the fact that  $g(\mu, \nu) \leq f_0(\tilde{x})$  holds for every feasible point  $\tilde{x}$ . The inequality (A.13) holds, but is vacuous, when  $g(\mu, \nu) = -\infty$ . The dual function gives a nontrivial lower bound on  $p^*$  only when  $\mu \succeq 0$  and  $(\mu, \nu) \in \mathbf{dom} g$ , i.e.,  $g(\mu, \nu) > -\infty$ . We refer to a pair  $(\mu, \nu)$  with  $\mu \succeq 0$  and  $(\mu, \nu) \in \mathbf{dom} g$  as *dual feasible*, for reasons explained in the following.

### The Lagrange Dual Problem

The optimization problem

$$\begin{aligned} & \text{maximize} && g(\mu, \nu) \\ & \text{subject to} && \mu \succeq 0 \end{aligned} \tag{A.14}$$

is called the *Lagrange dual problem* associated with the problem (A.12). In this context the original problem (A.12) is sometimes called the *primal problem*. The term *dual feasible*, to describe a pair  $(\mu, \nu)$  with  $\mu \succeq 0$  and  $g(\mu, \nu) > -\infty$ , now makes sense. It means, as the name implies, that  $(\mu, \nu)$  is feasible for the dual problem (A.14). We refer to  $(\mu^*, \nu^*)$  as *dual optimal* or *optimal Lagrange multipliers* if they are optimal for the problem (A.14).

The Lagrange dual problem (A.14) is a convex optimization problem, since the objective to be maximized is concave and the constraint is convex. This is the case whether or not the primal problem (A.12) is convex.

## Weak Duality

The optimal value of the Lagrange dual problem, which we denote  $d^*$ , is, by definition, the best lower bound on  $p^*$  that can be obtained from the Lagrange dual function. In particular, we have the simple but important inequality

$$d^* \leq p^*, \quad (\text{A.15})$$

which holds even if the original problem is not convex. This property is called *weak duality*, and we refer to the difference  $p^* - d^*$  as the *optimal duality gap* of the original problem, since it gives the gap between the optimal value of the primal problem and the best (i.e., greatest) lower bound on it that can be obtained from the Lagrange dual function. The optimal duality gap is always nonnegative.

## Strong Duality and Slater's Constraint Qualification

If the equality

$$d^* = p^* \quad (\text{A.16})$$

holds, i.e., the optimal duality gap is zero, then we say that *strong duality* holds. Strong duality does not, in general, hold. But if the primal problem (A.12) is convex, i.e., of the form

$$\begin{aligned} & \text{minimize} && f_0(x) \\ & \text{subject to} && f_i(x) \leq 0, \quad \text{for } i = 1, \dots, m, \\ & && Ax = b, \end{aligned} \quad (\text{A.17})$$

with  $f_0, \dots, f_m$  convex, we usually (but not always) have strong duality. There are many results that establish conditions on the problem, beyond convexity, under which strong duality holds. These conditions are called *constraint qualifications*. One simple constraint qualification is *Slater's condition*: There exists an  $x \in \text{relint} D$  (the interior of  $D$  relative to the affine hull of  $D$ ) such that

$$f_i(x) < 0 \quad \text{for } i = 1, \dots, m, \quad \text{and} \quad Ax = b. \quad (\text{A.18})$$

Such a point is sometimes called *strictly feasible*, since the inequality constraints hold with strict inequalities. Slater's theorem states that if Slater's condition holds and the problem is convex, then strong duality holds. Slater's condition implies not only strong duality for convex problems, but also that the dual optimal value is attained when  $d^* > -\infty$ , i.e., there exists a dual feasible  $(\mu^*, \nu^*)$  with  $g(\mu^*, \nu^*) = d^* = p^*$ .

## Applying Duality to a Specific Type of Convex Optimization Problem Involving Matrices

We now slightly modify (A.17) by using the matrix variable  $X$  for the unknown variable, which was previously the scalar variable  $x$ , and present the following development using ideas from [26]. One version of the problem with this matrix representation, on which we focus most in the thesis, is when  $f_1, \dots, f_m$  are linear operations and there is a psd constraint on  $X$ , i.e.,

$$\begin{aligned} & \text{minimize} && f_0(X) \\ & \text{subject to} && X \succeq 0, \\ & && \mathbf{Tr} A_i X = b_i, \quad \text{for } i = 1, \dots, p, \\ & && \mathbf{Tr} C_j X \leq d_j, \quad \text{for } j = 1, \dots, m, \end{aligned} \tag{A.19}$$

where  $A_1, \dots, A_p, C_1, \dots, C_m \in \mathbf{S}^n$ ,  $b_1, \dots, b_p, d_1, \dots, d_m \in \mathbb{R}$ , and  $f_0$  is convex. Note the similarity to (A.11) and that (A.19) is still a convex optimization problem.

Introducing the Lagrange multipliers  $\nu_1, \dots, \nu_p$  associated with the equality constraints,  $\mu_1, \dots, \mu_m$  associated with the inequality constraints, and the symmetric  $n \times n$  matrix  $Z$  associated with the matrix inequality  $X \succeq 0$  (which we write as  $-X \preceq 0$ ), the Lagrangian of problem (A.19) is then

$$L(X, Z, \nu, \mu) = f_0(X) - \mathbf{Tr} Z X + \sum_{i=1}^p \nu_i (\mathbf{Tr} A_i X - b_i) + \sum_{j=1}^m \mu_j (\mathbf{Tr} C_j X - d_j), \tag{A.20}$$

and the (Lagrangian) dual problem associated with the problem (A.19) is

$$\begin{aligned} & \text{maximize} && g(Z, \mu, \nu) \\ & \text{subject to} && Z \succeq 0, \quad \mu \succeq 0, \end{aligned} \tag{A.21}$$

where the dual function, i.e., the objective is given by  $g(Z, \mu, \nu) = \inf_X L(X, Z, \nu, \mu)$ .

Weak duality always holds for the dual problem (A.21): if  $Z$ ,  $\nu$ , and  $\mu$  are dual feasible, i.e.,  $Z \succeq 0$  and  $\mu \succeq 0$ , then the dual objective is a lower bound on the optimal value of problem (A.19). If (A.19) is strictly feasible, i.e., there exists an  $X \succ 0$  that satisfies the linear equalities and inequalities in (A.19), then strong duality holds: there exist  $Z^*$ ,  $\nu^*$ ,  $\mu^*$  that are optimal for the dual problem (A.21) with dual objective equal to the optimal value of the problem (A.19). Moreover, in some special cases (such as the strict convexity of the Lagrangian with respect to the primal variable  $X$ ), the optimal solution of the problem (A.19),  $X^* = \arg \min_X L(X, Z^*, \nu^*, \mu^*)$ , can be recovered from the dual optimal variables.

## Advantages of Dual Methods

When there is no duality gap, dual methods in general allow one to solve the dual problem, a related problem to the primal problem, and recover the solution of the primal problem. So one can use either the primal or the dual problem and find the same solution. The advantage is that sometimes solving the dual problem can be computationally easier than solving the primal. This may be due to the simpler form of the objective, the constraint set or both in the dual problem in comparison with the primal. In the thesis the reason we are using duality is that it results in a dual problem which has a much simpler constraint set than that of the primal problem.

## A.4 Matrix Calculus

### Numerator-layout Notation

Throughout the thesis we use the numerator-layout notation [42]. According to this notation, if  $Y = (y_{(i,j)})$  is an  $m \times n$  matrix and  $x$  is a scalar, then:

1.  $\partial Y / \partial x$  is an  $m \times n$  matrix with  $(i, j)^{\text{th}}$  element being  $\partial y_{(i,j)} / \partial x$ , and
2.  $\partial x / \partial Y$  is an  $n \times m$  matrix with  $(i, j)^{\text{th}}$  element being  $\partial x / \partial y_{(j,i)}$ .

### Chain Rule for Scalar by Scalar Derivative Involving Matrices

Let  $U(x)$  be a matrix and a function of a scalar  $x$ . Then the derivative of the scalar-valued function  $g(U)$  with respect to  $x$  in numerator-layout notation [43] is

$$\frac{\partial g(U)}{\partial x} = \mathbf{Tr} \left[ \frac{\partial g(U)}{\partial U} \frac{\partial U}{\partial x} \right] \quad (\text{A.22})$$

### Derivative of a Trace Function with respect to a Matrix

Now, we will derive the identity (again in numerator-layout notation)

$$\frac{\partial \mathbf{Tr}[BX^TAX]}{\partial X} = B^T X^T A^T + BX^T A. \quad (\text{A.23})$$

In numerator-layout notation, this is equivalent to deriving the identity

$$d\mathbf{Tr} [BX^TAX] = \mathbf{Tr} [(B^T X^T A^T + BX^T A)dX]. \quad (\text{A.24})$$

For that purpose, we will be using two properties of trace function:

1. It allows transposing, i.e.,  $\mathbf{Tr} [A^T] = \mathbf{Tr} [A]$ , and
2. It allows cyclic permutation, i.e.,  $\mathbf{Tr} [ABC] = \mathbf{Tr} [BCA] = \mathbf{Tr} [CAB]$ .



Now we proceed to the derivation:

$$\begin{aligned}
d\mathbf{Tr} [BX^T AX] &= d\mathbf{Tr} [AXBX^T] = \mathbf{Tr} [d(AXBX^T)] \\
&= \mathbf{Tr} [AXd(BX^T) + d(AX)BX^T] \\
&= \mathbf{Tr} [AXBd(X^T)] + \mathbf{Tr} [A(dX)BX^T] \\
&= \mathbf{Tr} [AXB(dX)^T] + \mathbf{Tr} [A(dX)BX^T] \\
&= \mathbf{Tr} [(AXB(dX)^T)^T] + \mathbf{Tr} [A(dX)BX^T] \\
&= \mathbf{Tr} [(dX)B^T X^T A^T] + \mathbf{Tr} [A(dX)BX^T] \\
&= \mathbf{Tr} [B^T X^T A^T (dX)] + \mathbf{Tr} [BX^T A (dX)] \\
&= \mathbf{Tr} [(B^T X^T A^T + BX^T A)dX].
\end{aligned}$$



# Appendix B

## Derivations for Subsection 4.3.2

In this appendix we derive the gradients (4.32, 4.33) of the dual objective  $g_{\text{sm}}$ , which is given below (and in (4.31)):

$$g_{\text{sm}}(Z, \mu) = \frac{1}{2} \|P_{\text{sm}}\|_F^2 + \lambda \|M_{\text{s}} \circ (D_{\text{s}} P_{\text{sm}})\|_F^2 - \text{Tr}[P_{\text{sm}} \tilde{G}] + \frac{1}{2} \|\hat{P}\|_F^2 - \mu^T d$$

Now, using the numerator layout, the fact that the trace function allows cyclic permutation, and the identity (A.23) from matrix calculus

$$\frac{\partial \text{Tr}[BX^TAX]}{\partial X} = B^T X^T A^T + BX^T A$$

as all explained in Section A.4, with the expansion formula for  $P_{\text{sm}}$  from (4.27), the symmetry property of  $\tilde{G}$ , the properties  $N_{i,j}^T = N_{j,i}$  and  $M_{i,j}^T = M_{j,i}$ , and the definitions  $I_{N \times N^2}^i = I_{N^2 \times N}^i$  and  $I_{1 \times N}^i = I_{N \times 1}^i$  as well, we will first find the gradient of each term of  $g_{\text{sm}}$  in (4.31) with respect to  $\tilde{G}$ . We will express each term in the form of a trace function of  $\tilde{G}$  before each gradient operation. Let us start with the first term.

$$\begin{aligned} \|P_{\text{sm}}\|_F^2 &= \text{Tr}[P_{\text{sm}}^T P_{\text{sm}}] = \text{Tr}\left[\left(\sum_{i=1}^N \sum_{j=1}^N N_{j,i}^T \tilde{G}^T M_{i,j}^T\right) \left(\sum_{k=1}^N \sum_{l=1}^N M_{k,l} \tilde{G} N_{l,k}\right)\right] \\ &= \sum_{i=1}^N \sum_{j=1}^N \sum_{k=1}^N \sum_{l=1}^N \text{Tr}\left[\underbrace{N_{l,k} N_{i,j}}_{N_{l,j} \text{ if } k=i, 0 \text{ otherwise}} \tilde{G}^T M_{j,i} M_{k,l} \tilde{G}\right] = \sum_{i=1}^N \sum_{j=1}^N \sum_{l=1}^N \text{Tr}\left[N_{l,j} \tilde{G}^T M_{j,i} M_{i,l} \tilde{G}\right] \\ \frac{\partial \text{Tr}[N_{l,j} \tilde{G}^T M_{j,i} M_{i,l} \tilde{G}]}{\partial \tilde{G}} &= N_{l,j}^T \tilde{G}^T M_{i,l}^T M_{j,i}^T + N_{l,j} \tilde{G}^T M_{j,i} M_{i,l} \\ &= N_{j,l} \tilde{G} M_{l,i} M_{i,j} + N_{l,j} \tilde{G} M_{j,i} M_{i,l} \end{aligned}$$

So, the derivative of the first term of  $g_{\text{sm}}$  in (4.31) with respect to  $\tilde{G}$  is

$$\begin{aligned} \frac{\partial \|P_{\text{sm}}\|_F^2}{\partial \tilde{G}} &= \sum_{i=1}^N \sum_{j=1}^N \sum_{l=1}^N [N_{j,l} \tilde{G} M_{l,i} M_{i,j} + N_{l,j} \tilde{G} M_{j,i} M_{i,l}] \\ &= 2 \sum_{i=1}^N \sum_{j=1}^N \sum_{l=1}^N N_{j,l} \tilde{G} M_{l,i} M_{i,j}. \end{aligned} \tag{B.1}$$

Let us now derive for the second term:

$$\begin{aligned}
\|M_s \circ (D_s P_{\text{sm}})\|_F^2 &= \|D_{s1} \text{vec}(P_{\text{sm}})\|_F^2 = \left\| \sum_{i=1}^N D_{s1} I_{N^2 \times N}^i \overbrace{P_{\text{sm}}}^{M_{k,j} \tilde{G} N_{j,k}} I_{N \times 1}^i \right\|_F^2 \\
&= \left\| \sum_{i=1}^N \sum_{k=1}^N \sum_{j=1}^N D_{s1} I_{N^2 \times N}^i M_{k,j} \tilde{G} \overbrace{N_{j,k} I_{N \times 1}^i}^{I_{N \times 1}^j \text{ if } i=k, 0 \text{ otherwise}} \right\|_F^2 = \left\| \sum_{i=1}^N \sum_{j=1}^N D_{s1} I_{N^2 \times N}^i M_{i,j} \tilde{G} I_{N \times 1}^j \right\|_F^2 \\
&= \sum_{i=1}^N \sum_{j=1}^N \sum_{k=1}^N \sum_{l=1}^N \text{Tr} \left[ \underbrace{I_{N \times 1}^l I_{N \times 1}^{jT}}_{N_{l,j}} \tilde{G}^T M_{i,j}^T \underbrace{I_{N^2 \times N}^i D_{s1}^T D_{s1} I_{N^2 \times N}^k}_{R_{i,k}} M_{k,l} \tilde{G} \right] \\
&= \sum_{i=1}^N \sum_{j=1}^N \sum_{k=1}^N \sum_{l=1}^N \text{Tr} \left[ N_{l,j} \tilde{G}^T M_{j,i} R_{i,k} M_{k,l} \tilde{G} \right] \\
\frac{\partial \text{Tr}[N_{l,j} \tilde{G}^T M_{j,i} R_{i,k} M_{k,l} \tilde{G}]}{\partial \tilde{G}} &= N_{l,j}^T \tilde{G}^T M_{k,l}^T R_{i,k}^T M_{j,i}^T + N_{l,j} \tilde{G}^T M_{j,i} R_{i,k} M_{k,l} \\
&= N_{j,l} \tilde{G} M_{l,k} R_{k,i} M_{i,j} + N_{l,j} \tilde{G}^T M_{j,i} R_{i,k} M_{k,l}
\end{aligned}$$

So, the derivative of the second term of  $g_{\text{sm}}$  in (4.31) with respect to  $\tilde{G}$  is

$$\begin{aligned}
\frac{\partial \|M_s \circ (D_s P_{\text{sm}})\|_F^2}{\partial \tilde{G}} &= \sum_{i=1}^N \sum_{j=1}^N \sum_{k=1}^N \sum_{l=1}^N \left[ N_{j,l} \tilde{G} M_{l,k} R_{k,i} M_{i,j} + N_{l,j} \tilde{G}^T M_{j,i} R_{i,k} M_{k,l} \right] \\
&= 2 \sum_{i=1}^N \sum_{j=1}^N \sum_{k=1}^N \sum_{l=1}^N N_{j,l} \tilde{G} M_{l,k} R_{k,i} M_{i,j}. \tag{B.2}
\end{aligned}$$

Now, we derive for the third term:

$$\begin{aligned}
\text{Tr}[P_{\text{sm}} \tilde{G}] &= \text{Tr} \sum_{i=1}^n \sum_{j=1}^n M_{i,j} \tilde{G} N_{j,i} \tilde{G} = \sum_{i=1}^n \sum_{j=1}^n \text{Tr} \left[ M_{i,j} \tilde{G}^T N_{j,i} \tilde{G} \right] \\
\frac{\partial \text{Tr}[M_{i,j} \tilde{G}^T N_{j,i} \tilde{G}]}{\partial \tilde{G}} &= M_{i,j}^T \tilde{G}^T N_{j,i}^T + M_{i,j} \tilde{G}^T N_{j,i} = M_{j,i} \tilde{G} N_{i,j} + M_{i,j} \tilde{G} N_{j,i}
\end{aligned}$$

So, the derivative of the third term of  $g_{\text{sm}}$  in (4.31) with respect to  $\tilde{G}$  is

$$\frac{\partial \text{Tr}[P_{\text{sm}} \tilde{G}]}{\partial \tilde{G}} = \sum_{i=1}^N \sum_{j=1}^N \left[ M_{j,i} \tilde{G} N_{i,j} + M_{i,j} \tilde{G} N_{j,i} \right] = 2 \sum_{i=1}^N \sum_{j=1}^N M_{i,j} \tilde{G} N_{j,i}. \tag{B.3}$$

Now we can combine the results (B.1), (B.2), and (B.3) to write down the gra-

dient of the dual function  $g_{\text{sm}}$  in (4.31) with respect to  $\tilde{G}$ :

$$\begin{aligned} \frac{\partial g_{\text{sm}}}{\partial \tilde{G}} &= \frac{1}{2} \cdot 2 \sum_{i=1}^N \sum_{j=1}^N \sum_{l=1}^N N_{j,l} \tilde{G} M_{l,i} M_{i,j} \\ &\quad + \lambda \cdot 2 \sum_{i=1}^N \sum_{j=1}^N \sum_{k=1}^N \sum_{l=1}^N N_{j,l} \tilde{G} M_{l,k} R_{k,i} M_{i,j} - 2 \sum_{i=1}^N \sum_{j=1}^N M_{i,j} \tilde{G} N_{j,i} \\ &= \sum_{i=1}^N \sum_{j=1}^N \left\{ -2 M_{i,j} \tilde{G} N_{j,i} + \sum_{l=1}^N \left[ N_{j,l} \tilde{G} \left( M_{l,i} + 2\lambda \sum_{k=1}^N M_{l,k} R_{k,i} \right) M_{i,j} \right] \right\}, \end{aligned}$$

which we can simplify to the following form if we denote  $E_{i,l}(\lambda) = M_{l,i}(\lambda) + 2\lambda \sum_{k=1}^N M_{l,k}(\lambda) R_{k,i}$ :

$$\frac{\partial g_{\text{sm}}}{\partial \tilde{G}} = \sum_{i=1}^N \sum_{j=1}^N \left( -2 M_{i,j} \tilde{G} N_{j,i} + \sum_{l=1}^N N_{j,l} \tilde{G} E_{i,l} M_{i,j} \right). \quad (\text{B.4})$$

To reduce the complexity of the calculation of  $\partial g_{\text{sm}} / \partial \tilde{G}$  in (B.4) from  $O(N^3)$  to much less, we will do a final trick before proceeding to finding the gradients of  $g_{\text{sm}}$  with respect to  $Z$  and  $\mu$ . We use once again the property that  $\text{vec}(ABC) = (C^T \otimes A) \text{vec}(B)$  where  $\otimes$  is the Kronecker product, to transform (B.4) into

$$\begin{aligned} \frac{\partial g_{\text{sm}}}{\partial \tilde{G}} &= \text{mat} \left\{ \text{vec} \left[ \frac{\partial g_{\text{sm}}}{\partial \tilde{G}} \right] \right\} \\ &= \text{mat} \left\{ \text{vec} \left[ \sum_{i=1}^N \sum_{j=1}^N \left( -2 M_{i,j} \tilde{G} N_{j,i} + \sum_{l=1}^N N_{j,l} \tilde{G} E_{i,l} M_{i,j} \right) \right] \right\} \\ &= \text{mat} \left\{ \underbrace{\left[ \sum_{i=1}^N \sum_{j=1}^N \left( -2 [N_{j,i}^T \otimes M_{i,j}] + \sum_{l=1}^N [M_{i,j}^T E_{i,l}^T \otimes N_{j,l}] \right) \right]}_{M_{\tilde{G}}(\lambda)} \text{vec}(\tilde{G}) \right\} \quad (\text{B.5}) \end{aligned}$$

$$= \text{mat} \left\{ M_{\tilde{G}} \text{vec}(\tilde{G}) \right\}, \quad (\text{B.6})$$

which simplifies the operation needed to compute  $\partial g_{\text{sm}} / \partial \tilde{G}$  significantly. We will use this expression in (B.6) for this gradient from now on.

Now, using the chain rule exactly as in the previous subsection, we can finally proceed to find the gradients of  $g_{\text{sm}}$  in (4.31) with respect to  $Z$  and  $\mu$ .

Since  $\tilde{G}$  is a linear function of  $Z$  and  $C(\mu)$  and they appear in  $g_{\text{sm}}$  via only  $\tilde{G}$ , the gradients of  $g_{\text{sm}}$  with respect to  $Z$  and  $C(\mu)$  can be found by a simple chain rule:

$$\frac{\partial g_{\text{sm}}}{\partial Z} = \frac{\partial g_{\text{sm}}}{\partial \tilde{G}} \frac{\partial \tilde{G}}{\partial Z} = \left( \text{mat} \left\{ M_{\tilde{G}} \text{vec}(\tilde{G}) \right\} \right) I = \text{mat} \left\{ M_{\tilde{G}} \text{vec}(\tilde{G}) \right\}, \quad (\text{B.7})$$

$$\frac{\partial g_{\text{sm}}}{\partial C(\mu)} = \frac{\partial g_{\text{sm}}}{\partial \tilde{G}} \frac{\partial \tilde{G}}{\partial C(\mu)} = \left( \text{mat} \left\{ M_{\tilde{G}} \text{vec}(\tilde{G}) \right\} \right) (-I) = -\text{mat} \left\{ M_{\tilde{G}} \text{vec}(\tilde{G}) \right\}. \quad (\text{B.8})$$

Since  $\mu_j$ 's are also involved in the dot product  $\mu^T d$  in  $g_{\text{sm}}$ , this dot product should be accounted for while calculating the gradient of  $g_{\text{sm}}$  with respect to  $\mu_j$ . Also using the chain rule (A.22) for scalar by scalar derivative involving matrices

$$\frac{\partial g(U)}{\partial x} = \mathbf{Tr} \left[ \frac{\partial g(U)}{\partial U} \frac{\partial U}{\partial x} \right],$$

this gradient is

$$\begin{aligned} \frac{\partial g_{\text{sm}}}{\partial \mu_j} &= \mathbf{Tr} \left[ \frac{\partial g_{\text{sm}}}{\partial C(\mu)} \frac{\partial C(\mu)}{\partial \mu_j} \right] + \frac{\partial(-\mu^T d)}{\partial \mu_j} = \mathbf{Tr} \left[ -\text{mat} \left\{ M_{\tilde{G}} \text{vec}(\tilde{G}) \right\} C_j^T \right] - d_j \\ &= -\mathbf{Tr} \left[ \text{mat} \left\{ M_{\tilde{G}} \text{vec}(\tilde{G}) \right\} C_j \right], \end{aligned} \quad (\text{B.9})$$

as  $d_j=0$  for all  $j=1, \dots, N(N-1)$ .

# Bibliography

- [1] R. M. Freund and S. Mizuno, “Interior point methods: Current status and future directions,” *OPTIMA*, vol. 51, pp. 1-9, Oct. 1996.
- [2] H. Hotelling, “Analysis of a complex of statistical variables into principal components,” *Journal of Educational Psychology*, vol. 24, no. 6, pp. 417-441, Sep. 1933.
- [3] M. E. Tippin and C. M. Bishop, “Probabilistic principal component analysis,” *Journal of the Royal Statistical Society, Series B (Statistical Methodology)*, vol. 61, no. 3, pp. 611-622, Sep. 1999.
- [4] T. W. Anderson, “Asymptotic theory for principal component analysis,” *Annals of Mathematical Statistics*, vol. 34, no. 1, pp. 122-148, Mar. 1963.
- [5] A. Basilevsky, *Statistical Factor Analysis and Related Methods: Theory and Applications*, New York: John Wiley and Sons, 1994.
- [6] D. M. Malioutov, “Approximate inference in Gaussian graphical models,” Ph.D. thesis, Massachusetts Institute of Technology, May 2008.
- [7] A. P. Dempster “Covariance selection,” *Biometrics*, vol. 28, no. 1, pp. 157-175, Mar. 1972.
- [8] T. P. Speed and H. T. Kiiveri, “Gaussian Markov distributions over finite graphs,” *The Annals of Statistics*, vol. 14, no. 1, pp. 138-150, Mar. 1986.
- [9] N. Meinshausen and P. Bühlmann, “High dimensional graphs and variable selection with the Lasso,” *The Annals of Statistics*, vol. 34, no. 3, pp. 1436-1462, Aug. 2006.
- [10] D. Bickson, D. Dolev, and E. Yom-Tov, “A Gaussian belief propagation for large scale support vector machines,” Technical Report, Hebrew University, 2007.

- [11] D. Bickson, D. Dolev, and E. Yom-Tov, "Solving large scale kernel ridge regression using a Gaussian belief propagation solver," *NIPS Workshop of Efficient Machine Learning*, 2007.
- [12] P. O. Vontobel, "Interior-point algorithms for linear-programming decoding," *Proc. Information Theory and its Applications Workshop*, 2008.
- [13] C. E. Rasmussen and C. K. Williams, *Gaussian Processes for Machine Learning*, MIT press, 2006.
- [14] R. Chellappa and S. Chatterjee, "Classification of textures using Gaussian Markov random fields," *IEEE Transactions on Acoustics Speech and Signal Processing*, vol. 33, no. 4, pp. 959-963, Aug. 1985.
- [15] R. Chellappa and A. K. Jain, *Markov Random Fields: Theory and Application*, Boston: Academic Press, 1993.
- [16] N. Cressie, *Statistics for Spatial Data*, New York: John Wiley and Sons, 1993.
- [17] A. Dobra, B. Jones, C. Hans, J. Nevins, and M. West, "Sparse graphical models for exploring gene expression data," *Journal of Multivariate Analysis, special issue on Multivariate Methods in Genomic Data Analysis*, vol. 90, no. 1, pp. 196-212, Jul. 2004.
- [18] C. Moallemi and B. Van Roy, "Consensus propagation," *IEEE Transactions on Information Theory*, vol. 52, no. 11, pp. 4753-4766, Nov. 2006.
- [19] H. Rue and L. Held, *Gaussian Markov Random Fields: Theory and Applications*, Boca Raton: Chapman and Hall, CRC, 2005.
- [20] I. T. Jolliffe, *Principal Component Analysis, 2nd ed.*, New York: Springer-Verlag, 2002.
- [21] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, New York: Springer-Verlag, 2001.
- [22] A. Ben-Tal and A. Nemirovski, *Lectures on Modern Convex Optimization: Analysis, Algorithms, and Engineering Applications*, Philadelphia: SIAM, 2001.
- [23] R. H. Tütüncü, K. C. Toh, and M. J. Todd, "Solving semidefinite-quadratic-linear programs using SDPT3," *Mathematical Programming, Series B*, vol. 95, no. 2, pp. 189-217, 2003.



- [24] Y. E. Nesterov, "A method of solving a convex programming problem with convergence rate of  $\mathcal{O}(1/k^2)$ ," *Soviet Math. Dokl.*, vol. 27, no. 2, pp. 372-376, 1983.
- [25] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM Journal on Imaging Sciences*, vol. 2, no. 1, pp. 183-202, Mar. 2009.
- [26] S. Boyd and L. Xiao, "Least-squares covariance matrix adjustment," *SIAM Journal on Matrix Analysis and Applications*, vol. 27, no. 2, pp. 532-546, Nov. 2005.
- [27] N. Higham, "Computing the nearest correlation matrix - a problem from finance," *IMA Journal of Numerical Analysis*, vol. 22, no. 3, pp. 329-343, Jul. 2002.
- [28] D. M. Malioutov, "Smooth isotonic covariances," *IEEE Workshop on Statistical Signal Processing*, Jun. 2011.
- [29] D. M. Malioutov, M. Cetin, and A. A. Corum, "Smooth and monotone covariance regularization," submitted to *Neural Information Processing Systems Workshop*, 2012.
- [30] P. J. Bickel and E. Levina, "Regularized estimation of large covariance matrices," *Annals of Statistics*, vol. 36, no. 1, pp. 199-227, 2008.
- [31] O. Ledoit and M. Wolf, "A well-conditioned estimator for large-dimensional covariance matrices," *Journal of Multivariate Analysis*, vol. 88, no. 2, pp. 365-411, Feb. 2004.
- [32] N. El Karoui, "Spectrum estimation for large dimensional covariance matrices using random matrix theory," *Annals of Statistics*, vol. 36, no. 6, pp. 2757-2790, 2008.
- [33] S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge University Press, 2004.
- [34] H. Markowitz, "Portfolio selection," *Journal of Finance*, vol. 7, no. 1, pp. 77-91, Mar. 1952.
- [35] D. M. Malioutov, "A sparse signal reconstruction perspective for source localization with sensor arrays," Master's thesis, Massachusetts Institute of Technology, Jul. 2003.

- [36] S. Mallat, G. Papanicolaou, and Z. Zhang, “Adaptive covariance estimation of locally stationary processes,” *Annals of Statistics*, vol. 26, no. 1, pp. 1-47, Feb. 1998.
- [37] D. M. Malioutov, J. K. Johnson, M. J. Choi, and A. S. Willsky, “Low-rank variance approximation in GMRF models: Single and multi-scale approaches,” *IEEE Transactions on Signal Processing*, vol. 56, no. 10, pp. 4621-4634, Oct. 2008.
- [38] T. Robertson, F. T. Wright, and R. L. Dykstra, *Order Restricted Statistical Inference*, New York: John Wiley and Sons, Jun. 1998.
- [39] S. Becker, J. Bobin, and E. J. Candès, “NESTA: A fast and accurate first-order method for sparse recovery,” *SIAM Journal on Imaging Sciences*, vol. 4, no. 1, pp. 1-39, Jan. 2011.
- [40] E. Derman and I. Kani, “The volatility smile and its implied tree,” *Quantitative Strategies Research Notes*, Goldman Sachs, Jan. 1994.
- [41] T. Bollerslev, “Generalized autoregressive conditional heteroskedasticity,” *Journal of Econometrics*, vol. 31, no. 3, pp. 307-327, 1986.
- [42] T. P. Minka, “Old and new matrix algebra useful for statistics,” MIT Media Lab note, Dec. 2000. Available: <http://research.microsoft.com/en-us/um/people/minka/papers/matrix/minka-matrix.pdf>
- [43] K. B. Petersen and M. S. Pedersen, “The matrix cookbook,” Nov. 2008. Available: <http://orion.uwaterloo.ca/~hwolkowi/matrixcookbook.pdf>
- [44] R. D. C. Monteiro, “First- and second-order methods for semidefinite programming,” *Mathematical Programming, Series B*, vol. 97, pp. 209-244, May 2003.
- [45] G. Lan, Z. Lu, and R.D.C. Monteiro, “Primal-dual first-order methods with  $\mathcal{O}(1/\epsilon)$  iteration-complexity for cone programming,” *Mathematical Programming, Series A*, vol. 126, no. 1, pp. 1-29, 2011.
- [46] G. Lan and R.D.C. Monteiro, “Iteration-complexity of first-order penalty methods for convex programming,” submitted to *Mathematical Programming*, 2012. Forthcoming, Online first, DOI: 10.1007/s10107-012-0588-x.
- [47] G. Lan and R.D.C. Monteiro, “Iteration-complexity of first-order augmented Lagrangian methods for convex programming,” submitted to *Mathematical Programming*, Jul. 2009.

- [48] R.D.C. Monteiro and B. F. Svaiter, “An accelerated hybrid proximal extra-gradient method for convex optimization and its implications to second-order methods,” submitted to *SIAM Journal on Optimization*, May 2011.
- [49] R.D.C. Monteiro, C. Ortiz, and B. F. Svaiter, “Implementation of a block-decomposition algorithm for solving large-scale conic semidefinite programming problems,” submitted to *SIAM Journal on Optimization*, May 2011.
- [50] D. Goldfarb and S. Ma, “Fast multiple splitting algorithms for convex optimization,” submitted to *SIAM Journal on Optimization*, 2009.
- [51] Y. Gao and D. F. Sun, “Calibrating least squares semidefinite programming with equality and inequality constraints,” *SIAM Journal on Matrix Analysis and Applications*, vol. 31, no. 3, pp. 1432-1457, Aug. 2009.